8-9-2020

Adaptive Trust Negotiation for Time-Critical Access to Sensitive Data

Eugene Sanzi
University of Connecticut - Storrs, eugene.sanzi@uconn.edu

Adaptive Trust Negotiation for Time-Critical Access to Sensitive Data

Eugene Nicholas Sanzi, Ph.D.
University of Connecticut, 2020

The security of an application's data is an important consideration when creating modern applications. Users requiring secure data access undergo an explicit pre-registration process where an electronic identity (username, X.509 certificate, etc.) and a method of laying claim to the identity (password, public/private key pair, etc.) are created. The user's authorization data is associated with the electronic identity. However, there are emergent situations where a user needs to access data where previous pre-registration is not possible because the future need for such data is unpredictable, such as an emergency room physician accessing the electronic health records (EHRs) of admitted patients. A process is needed where users (requestors such as medical personnel) make requests to the resource providers (controllers such as EHRs) in such a way that trust can be established automatically, allowing the requestor to obtain the necessary data quickly, securely, and safely.

The high-level focus of this dissertation is to present a trust negotiation framework that allows trust to be established with automated techniques by extending and combining trust negotiation and a new trust profile. Trust negotiation establishes trust by allowing a requestor and controller to alternate releasing secure credentials. The trust profile introduced in this dissertation is a complete history of the user's access to sensitive data.

The user chooses a subset of the trust profile and presents it to the controller during trust negotiation as proof that the user has been trusted to access sensitive data in the past. If the controller grants access to the user, the controller generates new credentials that the user receives and adds to the trust profile. The feasibility of this approach is demonstrated through a scenario in the healthcare industry, where healthcare professionals (doctors, nurses, insurance agents, public health officials, etc.) obtain authorization to healthcare data possessed by healthcare organizations, with whom there is no pre-existing relationship. We leverage health information exchange concepts, the Fast Healthcare Interoperability Resources (FHIR) standard, and the Connecticut Concussion Tracker app as the infrastructure within which trust profiles and trust negotiation are realized.

Adaptive Trust Negotiation for Time-Critical Access to Sensitive Data

Eugene Nicholas Sanzi

B.S., University of Connecticut, USA, 2011

M.S., University of Connecticut, USA 2019

A Dissertation

Submitted in Partial Fulfillment of the

Requirements for the Degree of

Doctor of Philosophy

At the University of Connecticut

2020

APPROVAL PAGE

Doctor of Philosophy Dissertation

Adaptive Trust Negotiation for Time-Critical Access to Sensitive Data

Presented By

Eugene Nicholas Sanzi, B.S., M.S.

Major Advisor: _____
Dr. Steven A. Demurjian

Associate Advisor: _____
Dr. Bing Wang

Associate Advisor: _____
Dr. Thomas Agresta

University of Connecticut

2020

ACKNOWLEDGEMENTS

First I would like to thank my major advisor, Prof. Steven Demurjian for guiding me throughout the entire process of earning my PhD, from the first moment I met with him to learn about the process until my defense and beyond. Your expertise and advice have allowed me to become the researcher I am today and I am glad I chose to meet with you all those years ago! I would also like to thank Dr. Thomas Agresta and Prof. Bing Wang for being my associate advisors and providing me excellent feedback and thoughts regarding my research.

Second I would like to thank Yaira. Your encouragement to never give up and the days we spent taking a break at the dairy bar and the mall made all the difference whenever things got busy. The times I spent with you and your family traveling through Puerto Rico, California, and Canada were some of the best times of my life! I would also like to thank Alberto, Antonio, Rishi, Timo, Ryan, and Brenden for creating a great environment in the lab during my first few semesters.

Next I would like to thank my parents and my sister, Alicia. Your support throughout my PhD and your visits while I was living in Storrs were always the highlight of my week, especially when I baked something new for you to try. I would like to thank my Papa for listening to my latest research and for teaching me about his camper's generator when I was barely old enough to walk, sparking my interest in engineering. I would like to thank my Grandma for her stories about her old horses, Daisy and Hoover, while I was learning to ride at UConn, and my Aunt Cheryl for always taking me to a nice restaurant whenever she visited in the summer. I would like to thank my Pépé and Mémé for all the good times I had at their house when I was young. Even though I never got the

chance to tell them about earning my PhD myself, I know they would've been proud. I would also like to thank the rest of my family for their support and encouragement while I was working on my PhD.

I would like to thank my friends for always listening to whatever problem I was having with my research and always being there to relax after day of hard work. To Dan, Jolene, and Demetrios the good times we spent together over the years from high school, to college, to now made this dissertation possible, and to Faye I hope someday we get to ride horses together.

Table of Contents

List of Tables

List of Figures

Chapter 1
Introduction

Modern computer systems are responsible for protecting a wide variety of secure data resources from malicious actors while also ensuring data availability to those who are sanctioned to access the data. Traditionally, access to data is determined via a process consisting of authentication, authorization, and registration. *Authentication* is the process of determining the user's identity. *Authorization* is the process assigning access rights (e.g., create or read a file) to the data. *Registration* refers to the process of generating the data necessary to authenticate and authorize a user. The user's identity consists of a *username*, which forms the basis of the identity, and a *secret password*, which is utilized as a means of claiming the identity. Creation of authorization data may be automated in simple cases, as in the case where a user only has a need to access data they create (e.g., email). A more complex case may require human intervention, such as a physician accessing electronic healthcare records (EHRs) from the local hospital's computer system. For this scheme to work, the username, password, and authorization data must be previously known to the computer system. The authentication process is simple and effective when it is known in advance that a specific user will have a need to access secure data residing in a known computer system. Conversely, the registration process is slow and inept when users may unexpectedly have a need to quickly access data from a computer system to which he/she has no previous existing relationship. To address this issue, trust negotiation

(Winsborough, Seamons, & Jones, 2000) provides a method for a set of user credentials to be released over an automated negotiation period.

Further complicating the processes of authentication, authorization and registration in a domain such as healthcare is the need to support health information exchange (HIE) so that medical providers treating patients can securely access multiple health information technology (HIT) systems (e.g., electronic health records (EHRs) patient portals, e-prescribing applications for medical providers and pharmacists, laboratory diagnostic systems, etc.) in both emergency and non-emergency situations to treat patients. This might include HIT systems to which the medical provider has not been given access previously. The increased interest in the secure sharing of data requires that methods for authenticating and authorizing users must become more sophisticated in order to support the needs of both the users and interactions with multiple HIT systems in the domain. The trust negotiation method outlined in this dissertation allows each HIT system (e.g., an EHR) to create and maintain its own criteria for data dissemination to users without the need for a central login system or pre-registration process. To support the trust negotiation process, each participating HIT system (for the user or the data source), must have a *controller* capability separate from that system in order to facilitate the interactions when the user without credentials requests access to one or more HIT systems they have not been authorized to use. The user in this case holds a record of each access to sensitive data, where each record is added to a collection that is referred to as a *trust profile*. This trust profile for a user contains records of access that have been accrued over time from multiple HIT systems that the user has been authorized to access. On the HIT system side of the trust negotiation process, the controller receives the trust profile from the user and is responsible for

interacting with the HIT system to decide if the requested access is allowable. When trust negotiation is successful, the controller creates a new entry in the trust profile for the user that can be utilized in future attempts to access sensitive data with any controller. Our proposed trust negotiation approach that is presented in this dissertation is integrated into the security model provided by the Fast Healthcare Interoperability Resources (FHIR) (HL7 International, 2020) standard, a health information exchange (HIE) standard created by HL7 to promote secure sharing of healthcare data among multiple health information technology (HIT) systems. Figure 1.1 illustrates the components needed to attain quick, automated trust negotiation for the release of secure data to a legitimate user. The leftmost side of Figure 1.1 indicates the components utilized by the user to obtain trust, whereas the rightmost side indicates the server-side components that check user credentials for validity, determine the level of access the user will obtain if any, and securely transfer the data to the user. Note that the user can be any medical stakeholder such as a physician, nurse, pharmacist, etc. While the examples in this dissertation are heavily dominated using the healthcare domain, the trust profile research that is presented as captured in the orange and blue boxes at the bottom of Figure 1.1 can be applied to any domain that is interested in tracking the history of users who are accessing highly sensitive data.

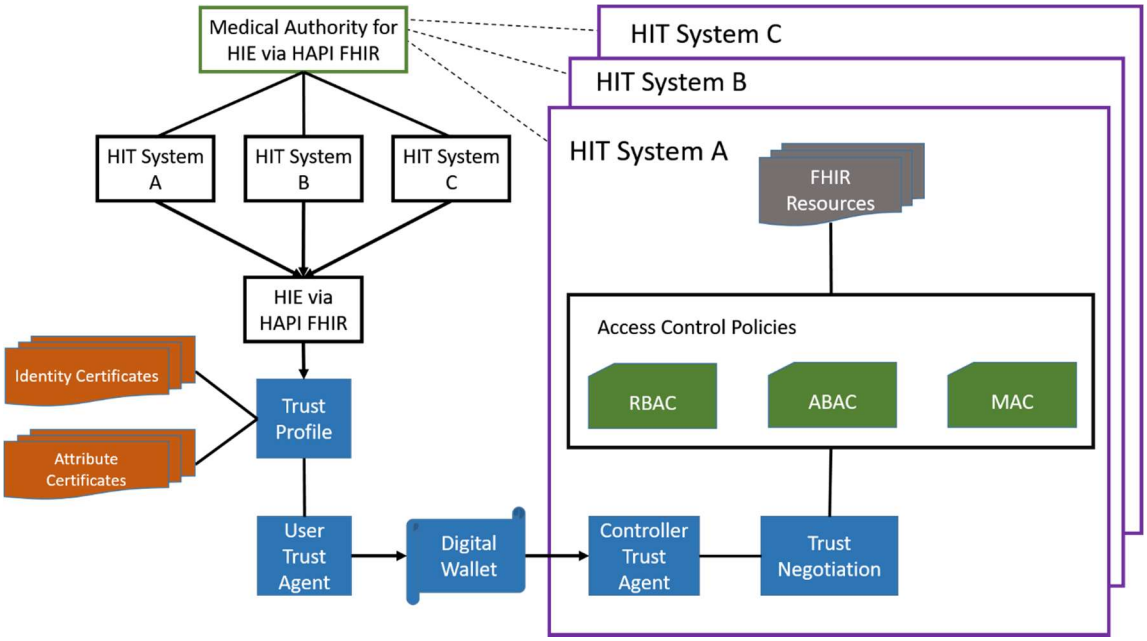

Figure 1.1. Interactions and Flow of Proposed Trust Negotiation Framework.

To understand all of the different interactions of trust negotiation, we explain all of the different components in Figure 1.1. This includes:

- The uppermost left corner a Figure 1.1 introduces the concept of a *Medical Authority*, which is an entity that promotes trust among the various HIT systems participating in trust negotiation, allowing them to trust each other's endorsements of a user's Trust Profile. The Medical Authority plays a major role in overseeing the process in interactions of all of the different components in trust negotiation.

- The upper left corner of Figure 1.1 has HIT Systems that the user has accessed data from that are accessible via a Medical Authority that front-ends a health information exchange layer that uses HAPI FHIR. These HIT systems endorse the

user by digitally signing the certificates that indicate the user has obtained access to the data listed in the Trust Profile.

- The Trust Profile in the lower left corner of Figure 1.1, represents a collection of the user's history of access to sensitive data, and includes: the User's Trust Agent, an autonomous actor that manages disclosure of the user's Trust Profile during the trust negotiation process; the Identity and Attribute certificates that encode the user's Trust Profile and add legitimacy to the Trust Profile by allowing the data to be verified and endorsed by a third party (a participating HIT System); and, a Digital Wallet, a subset of the user's Trust Profile that the user chooses to send with the request for sensitive data. The Digital Wallet is compiled by the user as an example of the user accessing similar sensitive data in the past.

- The Controller's Trust Agent in the lower right corner of Figure 1.1, is an autonomous actor that manages the disclosure of the controller's credentials and requests credentials from the user. The controller's Trust Negotiation component determines the type of credentials from the Trust Profile required to access the requested data, using Access Control Policies.

- The upper right corner of Figure 1.1 represents the health records that are available from the Medical Authority from one of the HIT Systems. Access to the health records/FHIR resources are via access control policies such as role-based access control, RBAC (Ferraiolo, Sandhu, Gavrila, Kuhn, & Chandramou, 2001), attribute

based access control ABAC (Hu, et al., 2014), and mandatory based access control MAC (Bell & La Padula, 1976).

Collectively, all of these components and their interactions provide the necessary infrastructure to allow controllers to generate requirements for the secure dissemination of sensitive data, request credentials from a user's Trust Profile, match those credentials to access control policies, transfer the requested data to the user, and generate new credentials for the user.

1.1. Motivation for Healthcare

The healthcare industry is increasingly interested in sharing healthcare data among medical providers (e.g., hospitals, clinics, pharmacies, public health officials, etc.) via HIE to enhance treatment and increase patient satisfaction (HealthIT.gov, 2014) (Kelly, 2013) (Mettler & Rohner, 2009). The healthcare industry is moving towards an approach to treatment where the patient sees a number of specialists in disparate subfields (e.g., general practitioners, cardiologists, oncologists, surgeons, dentists, psychiatrists, etc.) to treat a single patient. These healthcare professionals may not necessarily interact or know each other, but each must have complete access to the patient's healthcare record to form effective treatments. Healthcare workers in emergent care such as ER physicians or EMTs have a vested interest in obtaining patient data quickly, even if the patient's EHR data is located at multiple facilities. The availability of patient information across multiple HIT systems requires a Medical Authority that front ends a health information exchange layer supported by a standard such as FHIR as shown in Figure 1.1. The username/password

combination is insufficient in these scenarios as the preregistration process places too much of a burden on system administrators, requires the compilation of identity-based credentials from the physicians, and requires a lengthy vetting process. While a single, monolithic login platform with participation from all healthcare facilities would be sufficient in avoiding the drawbacks caused by the preregistration process, it creates a single point of failure, creates an extremely visible target for criminals, and diminishes the ability of controllers to control the dissemination of the Protected Health Information (PHI) they are charged with protecting under laws such as the Health Insurance Portability and Accountability Act (HIPAA) (U.S. Department of Health & Human Services, 2019).

Our trust profile based trust negotiation approach allows controllers (e.g., hospitals, health record banks, etc.) to automate the approval of the release of patient data by allowing them the flexibility to define the criteria that must be met before the data is released. Stakeholders such as physicians, hospitals, insurance companies, or public health officials are allowed to build trust amongst themselves through interactions over the course of a professional career. Users with a high amount of trust will find it relatively easy to request secure data, while users with a lower amount of trust may be required by controllers to meet stricter requirements to ensure that patient data is not leaked. The combination of expandable trust profiles with trust negotiation allows increased granularity in the credentials a user may present during the negotiation process, thus also increasing the granularity of the controller's security measures.

The increasing presence of mobile devices within the healthcare field provides additional difficulties as these devices may be lost or stolen, resulting in the exposure of sensitive patient data (Cisco, 2020). According to the Identity Theft Resource Center, a

non-profit organization that assists victims of identity theft, in 2019 the healthcare sector was found to have had the second most data breaches at 525 (Identity Theft Resource Center, 2020). An estimated 80% of physicians (Lewis, 2011) rely on mobile devices to access patient information from EHRs. Trust negotiation provides an extra layer of security by ensuring that the mobile device's user is deserving of trust. An adaptation of trust negotiation for mobile devices in the healthcare field was introduced in (Vawdrey, Sundelin, Seamons, & Knutson, 2003), which details a system for trust negotiation in healthcare while incorporating surrogate trust negotiation (Sundelin, July 2003), which addresses limited battery, slow computation, and unreliable networking issues in adapting trust negotiation to mobile devices. The possibility of patient Personally Identifiable Information (PII) and PHI being obtained via a stolen mobile device requires additional security parameters to ensure that patient data is not leaked. Trust negotiation has the potential to alleviate this concern by placing an insurmountable hurdle towards potential criminals; since they have not obtained a healthcare focused trust profile, it is impossible for them to obtain patient data from remote hospitals even with a stolen mobile healthcare device.

1.2. Motivation for Trust Profiling and Trust Negotiation

Many modern healthcare and financial industries are heavily data-driven and depend on the availability of critical data while also being extremely sensitive to improper data disclosure. In addition, there is a need for organizations operating within these fields to share data with others under certain conditions. Modern healthcare in particular requires teams of medical and non-medical professionals to treat a single patient, and each healthcare provider must have access to a patient's complete health record to form effective

treatments. Authorization data in both the healthcare and financial fields is often assigned manually in a slow registration process that guarantees that the data is released to the proper parties, but at the expense of time and the cost of needing an employee to manually perform the authorization. Authorizing individuals from other companies or fields may take even longer, as the identity and trustworthiness of the individual must be ascertained by the authorizer, which may slow down the process considerably.

The concepts of trust profiling and trust negotiation have the potential to greatly increase the speed at which authorization occurs while also properly restricting access only to those who should have access to the data by automating the assignment of permissions on the data, even to those who have no previous relationship with the data holder. Additionally, traditional authorization is often only able to produce an allow/deny decision, where the user may only be authorized to access all the data or none of it, despite the release of some forms of data (e.g., patient demographic information) being far less controlled and potentially damaging than others (e.g., patient mental health data). More nuanced forms of authorization, such as those outlined in this dissertation, allow for annotation on many different types of data and also allow the user to retrieve requested data to which they are authorized, while filtering more restricted data. Our approach also allows for the controller to automatically determine whether data must be added (e.g., including a required library so the requested data can be read), modified (e.g., translating an internal data representation into a data interchange standard), or removed (e.g., removing mental health data from a patient's EHR before transfer).

In support of these needs, our proposed trust negotiation approach seeks to allow for the integration of trust negotiation (Winsborough, Seamons, & Jones, 2000) and trust

profiles (Sanzi, Demurjian, Agresta, & Murphy, November 2016) with role-based access control (RBAC) (Ferraiolo, Sandhu, Gavrila, Kuhn, & Chandramou, 2001), attribute-based access control (ABAC) (Hu, et al., 2014), and mandatory access control (MAC) (Bell & La Padula, 1976). These three access control models allow us to leverage the trust negotiation procedure in a manner that allows for the automation of the assignment of permissions on data to the user and streamlines the negotiation of the release of portions of the user's Trust Profile in the form of a Digital Wallet as shown in the bottom of Figure 1.1. This Digital Wallet shown in the bottom center contains access history records that detail the Trust Profile owner's previous access to secured data. Portions of this data may be taken as credentials in the trust negotiation process that detail properties of the data accessed, such as the type of data access (e.g., patient drug list, financial report, etc.) or the identity of the patient listed in the record. When a controller receives a request for data in the bottom of Figure 1.1, the user's role in an RBAC scheme is sent along with the request so that the controller can ascertain the user's reason for requesting the data. For instance, a physician role in the healthcare field would be expected to access individual healthcare records, whereas a public health official would be expected to access more general data across many patients with regards to trends in public health (e.g., tracking concussions among high school athletes). ABAC, shown in the Access Control Policies in the center-right of Figure 1.1, allows the integration of credentials in the form of attributes from the Trust Profile to assist in determining whether the user has an access history that indicates experience and trustworthiness in handling the requested data, while also integrating MAC Access Control Policies to assign a security clearance level to the data.

The Trust Profile also enhances the security and efficiency of mobile devices by providing a scheme through which the user of the device can prove that the current user is authorized to access data. Mobile devices are becoming increasingly popular as a computing platform (Gartner, March 19, 2015), yet with increased popularity comes an increased focus on device security as more criminals consider mobile devices to be a prime target for data theft (Montopoli, 2013). Lost or stolen devices have the potential to allow criminals the opportunity to access sensitive data if proper security safeguards are not utilized. Stolen mobile devices may contain many types of sensitive data including: banking information, personally identifiable information (PII), protected health information (PHI), or allow the criminal the opportunity to access business or healthcare services. The addition of trust profiling and trust negotiation inserts an extra layer of security that allows any servers the device connects to the opportunity to determine that the current holder is a legitimate user by checking the user's access history.

1.3. A High-Level View of Our Approach

This dissertation seeks to create a method by which two entities whose identities are completely unknown to each other may initiate and fulfill requests for access to secure data or resources as illustrated in Figure 1.2. Figure 1.2 illustrates six levels through which the trust negotiation process is realized, from the local EHRs at the top of Figure 1.2 enhancing the level of trust for the user to remote HITs at the bottom of Figure 1.2 that users request sensitive PHI. The six levels are: Local EHR, User Devices, Trust Building, Trust Negotiation, Security Policies, and FHIR Resources from Remote HITs. There is a correspondence between the material presented in Figure 1.1 and the levels as given in Figure 1.2. Specifically, the orange and blue boxes at the bottom of Figure 1.1 correspond

to the Trust Building and Trust Negotiation levels in Figure 1.2 and these two levels are independent of the healthcare domain and could be leveraged and other domains that are interested in the secure access to information in a trusted manner. The User Devices present in the second level in Figure 1.2 allow healthcare professionals access to the Trust Building tools present in the Trust Building level, as well as offering connections to local EHRs and the ability to view sensitive PHI once authorized. The Medical Authority promotes trust among the healthcare organizations (HCOs), while the HCOs promote trust in an individual user's Trust Profile, allowing the user's access to trusted credentials. The Trust Building tools allow the User Devices to participate in Trust Negotiation where a user exchanges the credentials available from the Trust Building level to negotiate the release of a patient's healthcare data. The Trust Negotiation level encompasses the Credential Exchange, verification of the certificates through the trust building process (Profile Verification), and Release Actions performed if trust negotiation is successful, such as logging the transaction. The Security Policies level represents the security protocols that inform which credentials must be presented from the Trust Negotiation level to gain access to healthcare data provided by FHIR. The Security Policies filter the data provided by the FHIR Resources from Remote HITs illustrated in the bottom level in Figure 1.2 to ensure safe, secure sharing of sensitive healthcare data.

Access requests are facilitated through the Trust Negotiation level in a gradual exchange of increasingly sensitive credentials on the part of both the requestor and the controller until a mutual trust is established that: the requestor is qualified to access the requested resource, the requestor will handle sensitive resources in an appropriate manner, and the controller can be expected to provide the resource requested; or it is determined

that mutual trust cannot be established and the request fails. The credentials include a *trust profile*, a series of access records that encompasses the requestor's complete history of sensitive resource access including: when the request was made, the role the requestor held when the request was made, the requestor's affiliation status with healthcare providers, the resource requested, the controller the resource was requested from, the confidentiality of the resource, and the highest sensitivity level granted to the requestor from the controller. The Trust Profile is displayed in the lower left corner of Figure 1.1. During trust negotiation, the user's Trust Profile is gradually exchanged with the controller through the Digital Wallet, represented by the arrows flowing from the User Trust Agent to the Controller Trust Agent. The controller responds to a request for access to a resource from a user by using its Access Control Policies to determine the access history the user will need to provide through a subset of the Trust Profile to obtain access to the requested resource. The controller is illustrated in Figure 1.1 in the box encompassing the right side. The controller may also decide to take additional release actions depending on how well the requestor's presented trust profile fulfill its criteria. The controller may decide to: redact sensitive data, add data for interpretive purposes, modify data to arrange it into the correct format, or perform other functions such as dispatching audit notifications. Our approach allows each controller to determine its own Access Control Policies, giving the healthcare organization full control over its data disclosure requirements. The user also has full control over sensitive credentials released during the trust negotiation process and may choose to withhold sensitive trust profile entries until receiving additional assurances from the controller.

The *trust profile* is encoded in a series of X.509 Identity and Attribute Certificates (Housley, Polk, Ford, & Solo, 2002). These certificates are permanent representations of the user's access control history and are available to the user throughout his/her entire career. Should the user change employment, any certificates indicating current affiliation with the previous employer are revoked, but the history of sensitive record access may still be presented during trust negotiation. Each user possesses one Identity Certificate per controller from whom sensitive resource access has been granted. Each Identity Certificate may have one or more Attribute Certificates attached to it detailing the sensitive data that has been obtained. The Identity Certificates provide controllers with a verifiable method of determining that it has been presented by the owner of the trust profile by verifying ownership of the private key associated with the public key listed in the certificate. Attribute Certificates are attached to the Identity Certificates by verifying that the serial number and the issuer match the fields listed in the Identity Certificate. The digital signatures on the certificates, provided by the controller that granted access to the user, provide assurance that the information contained within the certificate is correct and has not been altered since the certificate was signed. The Trust Profile certificates may be held by the user in a local certificate store or stored with a *Trust Agent* that performs the trust negotiation process on the user's behalf.

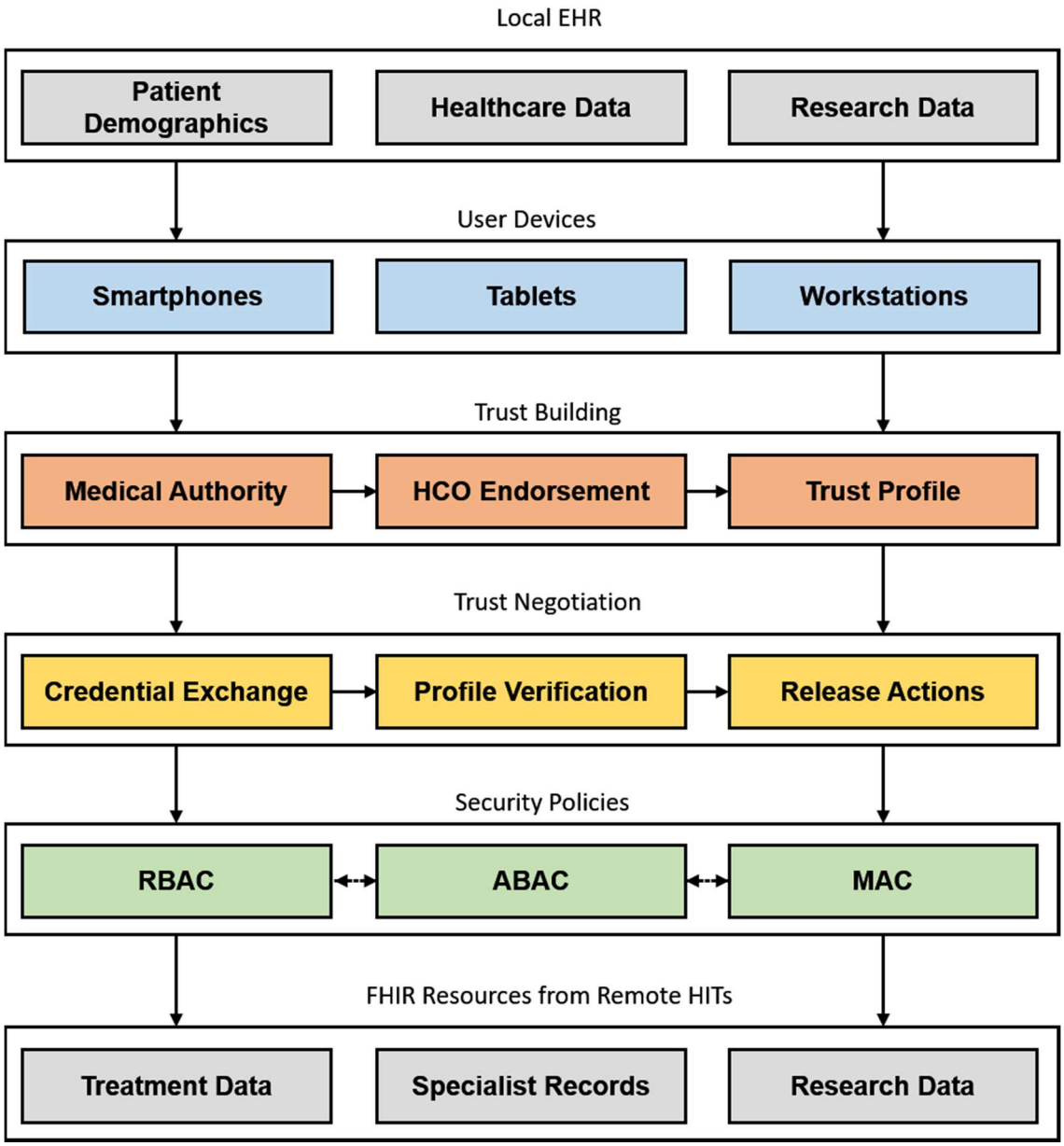

Figure 1.2. High-Level View of Trust Profile Supported Trust Negotiation.

The *Medical Authority* establishes trust between the healthcare organizations (HCOs) that own the HIT Systems by signing their controllers' Certificate Authority certificates, allowing the HIT Systems to sign the Trust Profile certificates as displayed in the Trust Building level of Figure 1.2. Medical Authorities are analogous to root certificate authorities – they promote trust among HCOs by acting as a mutually trusted third party,

symbolized by the arrow connecting medical authorities to HCOs in Figure 1.2. While certificate authorities are generally limited to providing assurance that the user has connected to the correct domain or verifying the domain owner's legal identity, the Medical Authority is responsible for also providing assurance that HCOs that can create trust profile entries are: producing accurate records, protecting their private keys appropriately, and enforcing a minimum standard on data disclosure. Each controller for an HCO must maintain a store of certificates belonging to medical authorities whose judgement is trusted by the administrator. During the inspection of certificates that represent entries in the requestor's Trust Profile, the digital signature on the certificates is inspected to ensure that the data within the certificate has not been altered. Then the certificate of the signer is inspected in the same manner. In most cases, this certificate will be a certificate belonging to another controller that the requestor has previously gained sensitive access to, with an entry indicating that the other controller has been authorized by a Medical Authority to produce Trust Profile certificates for its domain, and the Medical Authority's digital signature. If this certificate is valid, the certificate of the Medical Authority is retrieved and is checked against the controller's certificate store: if a matching certificate is found the trust profile certificates are valid. This establishes a chain of trust depicted in the Trust Building level of Figure 1.2. The Medical Authority establishes trust among HCOs, while the HCOs establish trust in each user's trust profile through a chain of digital signatures.

The system administrators can specify a series of requirements of the user's history that prove a need to access the data, responsibility in handling similar data, and allow access to secure data by applying RBAC, ABAC, MAC, etc., with a granularity encompassing health records, groups of health record data (demographic, insurance related,

etc.), and individual entries (a single MRI scan). This functionality is represented by the Security Policies of Figure 1.2. For instance, an administrator can indicate that general patient demographic information is available to requestors possessing the nurse or physician roles from other hospitals if they are able to present a credential indicating that they are presently employed by a healthcare organization recognized by a Medical Authority and have accessed healthcare data annotated with the lowest security level. Conversely, for more sensitive information such as a patient's complete EHR, the administrator can specify that: the requestor needs Trust Profile entries indicating that the requestor has accessed the same patient's healthcare record from other organizations and the controller must report the access for review by an auditor. This additional flexibility allows for the sharing of potentially sensitive healthcare data with healthcare professionals that have a need to access sensitive healthcare data who have proven to be trustworthy in the past, freeing administrators from a time-consuming manual vetting of a requestor's trustworthiness and allowing the release of healthcare data immediately in time-critical situations.

1.4. Research Objectives and Expected Contributions

The proposed solution for integrating trust negotiation and trust profiling has the following expected contributions.

A. Infrastructure Requirements to Promote Trust Among Organizations Participating in Trust Negotiation: This contribution will define a set of infrastructure requirements that organizations must provide to support the ability to establish implicit trust in the credentials generated and signed by one another and to enforce the trust

negotiation process, represented by the Trust Building process of Figure 1.2. This trust between organizations forms the basis of the Trust Profile, as the organizations must be sure that the presented credentials in the Trust Profile are valid to make an informed decision as to the trustworthiness of the requestor. The left side of Figure 1.1 represents the trust building network and encompasses the Medial Authority for HIE via HAPI FHIR, each HIT system, HIE via HAPI FHIR, and the Trust Profile's Identity and Attribute Certificates.

B. Integrated Trust Profile Model for Recording Complete Records of User Access to Sensitive Data: The contribution will include a format for the Trust Profile that provides the ability to record metadata describing sensitive data access and built in integrity checks during the Trust Negotiation level of Figure 1.2, allowing for the Trust Profile to be utilized as a credential with minimal communication required between participating organizations. These interactions occur between controllers and requestors. The model includes granular access control annotation of healthcare data stored in a HAPI FHIR server and the methods utilized to match trust profile access history data to the annotated FHIR data. This contribution provides the internal structure for the Identity and Attribute Certificates depicted on the left of Figure 1.1 and the process of exchanging Trust Profile credentials between the User Trust Agent and the Controller Trust Agent.

C. Dynamically Generated Adaptive Access Control Policies: This contribution is a combination of access control models RBAC, ABAC, and MAC that provide a method for organizations to utilize a set of security policies that can be dynamically adapted to the current request, located in the Security Policies level of Figure 1.2. Requirements for the release of data depend on: the role the requestor assumes during the negotiation, the type

of healthcare data requested, the portions of the Trust Profile released during negotiation, and the security annotations attached to the data resources. This contribution forms the structure and interactions between the Access Control Policies on the right side of Figure 1.1.

D. Trust Negotiation Development Framework: Contribution D provides the participating organization with all of the required steps and processes that are necessary to: define the infrastructure in support of Contribution A; support and implement the model for Contribution B; and, enforce a dynamic adaptive trust negotiation process in support of Contribution C. This contribution defines the interactions between the various components in a full, trust negotiation capable HIT system depicted on the right side of Figure 1.1. Contribution D provides the necessary links between the Trust Negotiation, Security Policies, and FHIR Resources from Remote HITs level in Figure 1.2.

This dissertation will examine each contribution's relevance towards trust profiles and trust negotiation in the healthcare field.

1.5. Research Progress to Date

In support of the proposed trust profile based trust negotiation, we summarize our 4 publications (4 published) and their contribution toward the dissertation: lead author directly related to the work are: 2 published refereed book chapters and 2 published refereed full conference articles; coauthor of 2 published journal articles; coauthor of 1 published refereed book chapter; and coauthor of 1 published refereed full conference article. The first papers focused on the overall concept and flow of a trust profile based trust negotiation process, as well as a network structure for disseminating trust throughout

the healthcare network (Sanzi & Demurjian, Identification and Adaptive Trust Negotiation in Interconnected Systems, May 2016). This work was expanded with a method of adapting trust profiles and trust negotiation to mobile devices including: description of the contents and set of interactions for trust profiles stored on mobile devices during trust negotiation; and detailed descriptions of the exchange of credentials between the user's mobile device, trust agents, and the controller; and the generation and storage of new trust profile credentials if the trust negotiation is successful (Sanzi, Demurjian, Agresta, & Murphy, November 2016).

- **Sanzi, E**. and S. Demurjian, "Identification and Adaptive Trust Negotiation in Interconnected Systems," in *Innovative Solutions for Access Control Management*, A. Malik, A. Anjum and B. Raza, Eds., IGI Global, May 2016, pp. 33-65.
- **Sanzi, E.**, S. Demurjian, T. Agresta and A. Murphy, "Trust Profiling to Enable Adaptive Trust Negotiation in Mobile Devices," in *Mobile Application Development, Usability, and Security*, S. Mukherja, Ed., IGI Global, November 2016, pp. 95-116.

Building upon these trust building concepts, we defined a formal model for the trust profile (Sanzi, Demurjian, & Billings, Integrating Trust Profiles, Trust Negotiation, and Attribute Based Access Control, 2017). This formal model provides a standardized structure for the trust profile, including a set of standardized attribute certificate structures to describe each access to sensitive healthcare data and a set of attribute certificates describing overall sensitivity levels and local affiliation with a healthcare organization as a whole.

- **Sanzi, E.**, Demurjian, S., & Billings, J. (2017). Integrating Trust Profiles, Trust Negotiation, and Attribute Based Access Control. *2017 5th IEEE International*

Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud) (pp. 177-184). San Francisco: IEEE. doi:10.1109/MobileCloud.2017.30

The next effort provides a methodology for incorporating trust profile based trust negotiation into the FHIR standard by providing a standard set of security objects. These security objects are generated based on a configuration set by a system administrator, as well as the patient described by the resource the security objects belong to, and adapts the security objects based on the user's role and the credentials chosen from the trust profile by the user during trust negotiation.

- Sanzi, E. and Demurjian, S., "Trust Profile Based Trust Negotiation for the FHIR Standard," *Proceedings of 9th International Conference on Data Science, Technologies, and Applications (DATA2020),* July 2020.

Other publications not directly related to the work are:

- Ziminski, T., Demurjian, S., **Sanzi, E.**, Baihan, M. and Agresta, T., "An Architectural Solution for Health Information Exchange," republished in *Virtual and Mobile Healthcare: Breakthroughs in Research and Practice,* Information Resources Management Association (USA), pp. 283-327, July 2020.
- Demurjian, S., **Sanzi, E.,** Agresta, T., and Yasnoff, W., "Multi-Level Security in Healthcare using a Lattice-Based Access Control Model," IGI *International Journal of Privacy and Health Information Management (IJPHIM)*, Vol. 7, No. 1, January-June 2019, pp. 80-102, IGI Global,
- Ziminski, T., Demurjian, S., **Sanzi, E.**, Baihan, M. and Agresta, T., "An Architectural Solution for Health Information Exchange," *International Journal of*

User-Driven Healthcare (IJUDH), Vol 6, No. 1, pp. 65-103, November 2016, IGI Global, https://www.igi-global.com/article/an-architectural-solution-for-health-information-exchange/181318

- Ziminski, T. B., Demurjian, S. A., **Sanzi, E**., & Agresta, T. (2016). Toward Integrating Healthcare Data and Systems: A Study of Architectural Alternatives. In T. Iyamu, & A. Tatnall (Eds.), *Maximizing Healthcare Delivery and Management through Technology Integration*, pp. 270-304. IGI Global. doi:10.4018/978-1-4666-9446-0.ch016
- Agresta, T., Demurjian, S., **Sanzi, E.**, DeStefano, J., Ward-Charlerie, S., Rusnak, R., & Tran, R. (2020). A Mobile Health Application for Medication Reconciliation using RxNorm and FHIR. Submitted to *The Fifth International Conference on Informatics and Assistive Technologies for Health-Care, Medical Support and Wellbeing (HEALTHINFO 2020).* Porto, Portugal.

1.6. Dissertation Outline

The remainder of the dissertation has 6 chapters. In Chapter 2, we detail background on the requirements for security in the healthcare field, relevant RBAC, ABAC, and MAC access control models utilized to restrict information access, supporting technologies for trust negotiation, and the FHIR standard/HAPI server implementation for supporting the trust profile infrastructure. Chapter 2 also briefly presents the Connecticut Concussion Tracker (CT^2) app that was created as the result of a new law passed in Connecticut (Connecticut General Assembly, 2015) requiring that concussions be tracked for kids between the ages of 7 to age 19 in public schools. In Chapter 3, we address: Contribution

A: Infrastructure Requirements to Promote Trust Among Organizations Participating in Trust Negotiation, which establishes how healthcare organizations and their affiliates connect on the network, establish communication protocols, and create a baseline of trust in the credentials themselves; Contribution B: Integrated Trust Profile Model for Recording Complete Records of User Access to Sensitive Data by detailing the identity and attribute certificates that form the trust profile; and, Contribution C: Dynamically Generated Adaptive Access Control Policies by defining integration of access control policies into attribute certificates. In Chapter 4, we primarily address Contribution B: Trust Profile for Recording Complete Records of User Access to Sensitive Data by describing the interactions between the requestor and the data controller that allows a gradual growth in trust to occur. Chapter 4 also addresses Contribution C: Dynamically Generated Adaptive Access Control Policies, by describing how the access control policies protect the underlying healthcare data and establishes the relationship between the access control policies and the trust establishment process through a defined formal model. Contribution C is illustrated utilizing a mobile health (mHealth) application for concussion management which leverages health information exchange concepts and a RESTful API as the infrastructure within which are trust profiles and trust negotiation are realized. In Chapter 5, we address Contribution C: Dynamically Generated Adaptive Access Control Policies by defining integration of access control policies with the controller and Contribution D: Adaptive Trust Design and Development Methodology by providing a framework for integrating dynamic and adaptive trust negotiation to existing healthcare services. Chapter 6 addresses Contribution C: Dynamically Generated Adaptive Access Control Policies with a detailed description of a controller implementation that utilizes the access control

policies to guard access to the CT^2 app prototype and Contribution D: Trust Negotiation Development Framework by including a detailed discussion of the prototype (CT^2) that has been developed to demonstrate unification of the trust profile concepts presented throughout this dissertation. Finally, Chapter 7 provides a summary of the contributions in the dissertation and a detailed discussion of potential future research directions and efforts.

Chapter 2
Background

This chapter describes background information necessary for the main contributions and supporting concepts utilized throughout the remainder of this dissertation. Section 2.1 has a brief overview of the state of trust and interoperability between healthcare systems and the standards that must be met to enable secure interoperability. Section 2.2 explains role-based access control (RBAC) (Ferraiolo, Sandhu, Gavrila, Kuhn, & Chandramou, 2001), attribute-based access control (ABAC) (Hu, et al., 2014), and mandatory access control (MAC) (Bell & La Padula, 1976) and their application to healthcare security. Section 2.3 provides a detailed explanation of the FHIR (HL7 International, 2020) standard and the HAPI server (HAPI FHIR, 2020) in support of the trust negotiation model. Section 2.4 concludes the background with a description of the Connecticut Concussion Tracker (CT^2) mobile app, a collaboration between the Departments of Physiology and Neurobiology, and Computer Science & Engineering at the University of Connecticut and Schools of Nursing and Medicine. The CT^2 app was created in support of a new Connecticut law passed to track concussions among students for grades kindergarten through high school (Connecticut General Assembly, 2015). The application will be utilized in Chapter 6 in order to incorporate the trust profile concepts of Chapters 3, 4, and 5 into a prototype in order to demonstrate how the entire trust profiling process works.

2.1. State of Trust and Interoperability

Modern healthcare treatment requires a large amount of data to function efficiently and effectively. A patient's healthcare record includes important data compiled through the use of various tests that are highly specialized and expensive, uncomfortable for the patient, and sometimes result in long waiting periods before the results of the test are available. Such tests may include: blood and tissue laboratory tests, X-Rays, EKGs, mammograms, MRI scans, etc. Increased specialization in the healthcare field has created a need for patients to travel between multiple healthcare providers, each performing one portion of the patient's treatment. Interoperability between the healthcare systems that the providers utilize to compile and store patients' healthcare records reduces stress to the patient and increases efficiency by allowing these teams of specialists to be informed of the patient's current condition, treatments performed by other specialists, and patient history that may inform current treatment. However, the current state of trust and interoperability within the healthcare field has created barriers to this process including incompatible electronic health record (EHR) formats and a lack of universal authentication and authorization methods.

The creation of the meaningful use guidelines (Centers for Disease Control and Prevention, 2018) has incentivized many EHR software packages such as OpenEMR (OpenEMR, 2020), Epic (Epic Systems Corporation, 2020), and WorldVistA (WorldVistA, 2020) to create new interoperability software. Although there has been improvements, these software solutions often still suffer from issues that limit their overall effectiveness. For example Epic's Care Everywhere (Yale New Haven Health; Yale Medical Group, 2020) creates an electronic exchange among healthcare providers but only operates between Epic's own EHR installations, specifically the different medical practices and office is associated with Yale New Haven health. One barrier to the adoption of

interoperability standards is the lack of a universally adopted standard for health information exchange (HIE). Health Level Seven International (HL7 International, 2019) (HL7) is an organization created to rectify this by creating and promoting standards for EHR interoperability. HL7's latest standard, Fast Healthcare Interoperability Resources (FHIR) (HL7 International, 2020) is quickly being adopted by EHRs to facilitate interoperability standards for the exchange of healthcare records. OpenEMR currently has in-progress work on FHIR integration, but the full implementation is not yet available.

Although interoperability among healthcare providers is an ongoing goal, there is yet to emerge a universal standard for healthcare authentication and authorization. Traditionally, authentication and authorization is performed manually by an employee by vetting the requestor's identity and assigning them a set of permissions on the requested data. One potential standard, OAuth2 (Hardt, 2012), is being utilized by large companies such as Facebook (Facebook, 2020), Google (Google, 2020), and Twitter (Twitter, Inc., 2020). OAuth2 has the potential to be both a universal authenticator and authorizer by offloading the authentication/authorization processes to a third party during a request, then sending authentication/authorization data back to the controller. However, OAuth suffers from some weaknesses. Although OAuth does have widespread adoption and strong industry support, it still requires that the user has been pre-registered before a request can be authenticated and only offloads the authentication process to a trusted third party. The adoption of a trust negotiation approach eliminates this weakness by allowing for trust to be negotiated at request time without needing any prior details and without requiring an identity to authenticate to.

2.2. Access Control

Role-based Access Control (RBAC) (Ferraiolo, Sandhu, Gavrila, Kuhn, & Chandramou, 2001) is an access control model that organizes permissions (read, write, execute, etc.) into collections referred to as roles. Each user in the access control model is assigned one or more roles and each role assigned to the user entitles him/her to the associated permissions. When the user requests access to an object protected under RBAC, the user chooses a role they have been previously assigned and the permissions associated with the role are matched against the requested object. If the role the user has chosen has sufficient permissions to access the object, the user obtains access; otherwise the user is denied access. RBAC is a popular access control model in the healthcare field due to its model matching the methods typically utilized to decide whether a healthcare organization's employee may access healthcare data (Fernández-Alemán, Señor, Lozoya, & Toval, 2013) (Alhaqbani & C., 2008). Common roles include: physicians, nurses, billing, administration, or receptionists. Data protected by RBAC in a healthcare setting includes: demographic data, blood pressure readings, X-Rays, appointment times, ICD10 codes, insurance information, etc. During treatment, a physician role may be assigned permissions to access the entirety of the patient's health record including demographics, treatment history, and tests, but would be denied access to sensitive records such as the patient's mental healthcare data. Likewise, the billing department's roles would not have access to the patient's full health record, but would have access to the patient's insurance info, ICD10 codes, and any other data required to properly bill the patient or the insurance company.

Mandatory Access Control (MAC) (Bell & La Padula, 1976) is constructed utilizing the concept of tagging information with a *sensitivity level* that represents

confidentiality of the data by defining different categories of information. Subjects under MAC are assigned a sensitivity level (clearance) based on the level of trust obtained. Likewise, objects protected by MAC are also assigned a sensitivity level (classification) based on their sensitivity. MAC's sensitivity model generally is comprised of four sensitivity levels: top secret (TS) > secret (S) > classified (C) > unclassified (U). In most MAC schemes, a user is permitted to read data that has been classified at or below the user's assigned clearance. A user is permitted to write data which is then classified at or above the user's assigned clearance to prevent sensitive data from leaking downwards. In the healthcare industry, a MAC model can be utilized to assign sensitivity levels to data, allowing for more granular permissions on healthcare data that may be more sensitive. A patient's demographic information may be classified as U, since the patient's name, phone number, and contact information may be considered low risk, allowing any user with a U clearance to request patient demographic information. However, a patient's drug list may be classified as S. Since a drug list may reveal a patient's conditions, it is considered more sensitive information. In this case, a user must have at least an S clearance to view the patient's drug list. The HL7 vocabulary (HL7 International, 2013) defines sensitivity levels summarized in Table 2.1 of low, moderate, normal, restricted, unrestricted and very restricted, depending on the perceived damage if the healthcare data was leaked. Note that we have published an article (Demurjian, Sanzi, Agresta, & Yasnoff, January-June 2019) on security of healthcare data utilizing lattice-based access control which is the parent of MAC.

U	unrestricted	This indicates that the information is not classified as sensitive such as publicly available information, e.g., business name, phone, email or physical address.
L	low	The information requires protection to maintain low sensitivity such as anonymized, pseudonymized, or non-PII such as HIPAA limited data
M	moderate	This is moderately sensitive information, which presents moderate risk of harm if disclosed without authorization.
N	normal	Non-stigmatizing health information, which presents typical risk of harm if disclosed without authorization.
R	restricted	Highly sensitive, for normal clinical care, potentially stigmatizing information with a high risk if disclosed without authorization.
V	very restricted	Extremely sensitive and likely stigmatizing health information that presents a very high risk if disclosed without authorization.

Table 2.1. HL7 Confidentiality Levels.

Attribute-based Access Control (ABAC) (Hu, et al., 2014) is an access control model where access to an object is dependent upon: a user's attributes, the object's attributes, and a policy engine that can match the involved attributes, a set of rules, and external environmental conditions. The user possesses a set of attributes that describe the user, such as: licensing, roles, patient lists, or work schedule. The objects possess their own set of attributes such as: patient ID, primary physician, or last editor. The policy engine contains a list of programmed access control rules it incorporates into its decision making process. During a user request for access, the policy engine reads the user attributes, the object's attributes, other environmental conditions such as the current time, and the access control rules. Using the access control rules, the policy engine attempts to match the user's attributes, the object's attributes, and the environmental conditions in a manner that satisfies the access control rules. If this is successful, the user is granted the requested

access to the object. The policy engine consists of the policy decision point (PDP) and the policy enforcement point (PEP). The PDP is responsible for making a decision as to whether the access control rules are satisfied while the PEP is responsible for ensuring that the decision made by the PDP is enforced. By restricting access based on a set of attributes, ABAC allows us to restrict or grant access to objects based on credentials that need not include the user's actual identity.

ABAC's core model seen in Figure 2.1, redrawn from (Hu, et al., 2014), contains an example related to the healthcare domain. In Figure 2.1, Physician Tom, located to the left side of the figure, possesses the attributes: (ROLE: PHYSICIAN and PRIMARY_PHYSICIAN for PATIENT_ID: 1). The resource objects, the patient's healthcare records located in the right side of the figure, are annotated with the attributes: (PATIENT_ID, PRIMARY_PHYSICIAN, APPOINTMENT_TIME). The listed access control rules located in the bottom of the figure indicate that read/write access is granted to subjects that are the primary physician of the patient and the patient must have an appointment time matching the current time. Read access is granted to any subject who is either a nurse or a primary physician. Should Physician Tom request access to the health record of Patient 1, the PDP will read the access control rules, Physician Tom's attributes, the attributes for Patient 1's health record, and the current time. Physician Tom's attributes indicate that he is both a physician and the primary physician for Patient 1. The record indicates that it is the record for Patient 1 and that the patient has an appointment at the current time. Since these attributes match the first rule, the PDP decides that Physician Tom should be granted read/write access to the health record. The PEP ensures that Physician Tom has read write access to the healthcare record. Suppose that the user is a

nurse. In this case, the second access control rule is invoked, the PDP decides to grant read access, and the PEP will enforce the read only access on the requested healthcare record.

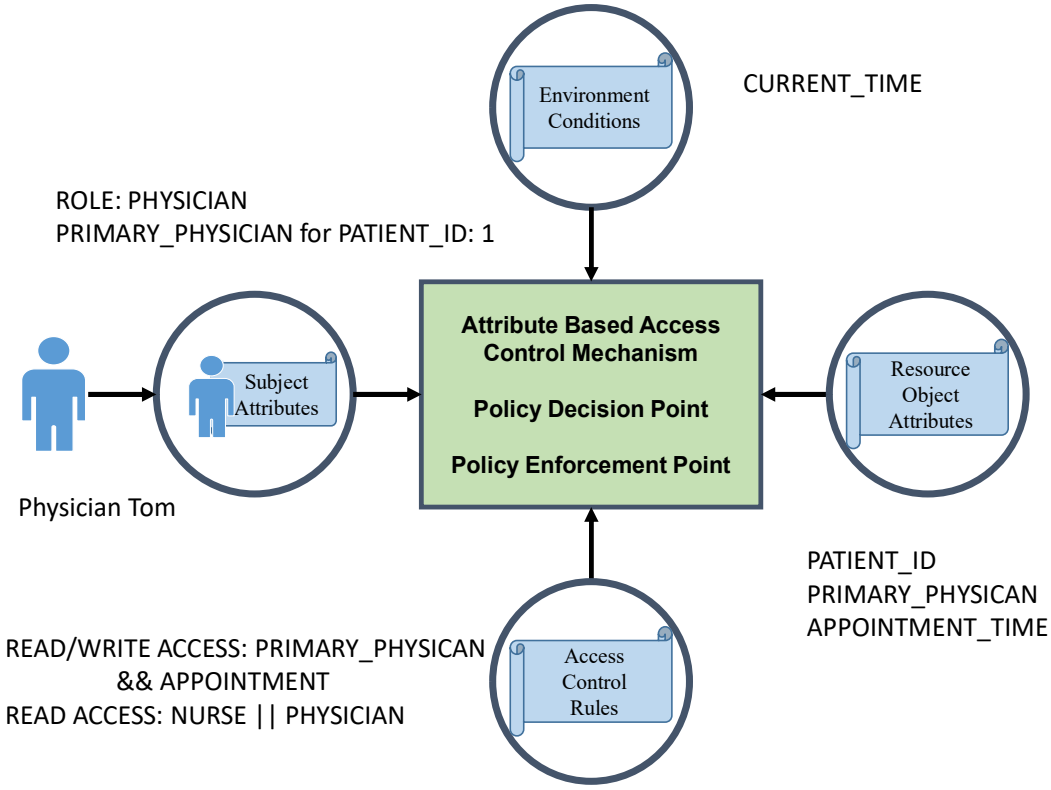

Figure 2.1. The ABAC Model.

2.3. Fast Healthcare Interoperability Resources (FHIR)

The Fast Healthcare Interoperability Resources (FHIR) specification (FHIR DSTU2, 2015) is a standards framework created by the Health Level Seven (HL7) organization (HL7, 2011) with the intention of providing easier and quicker implementation of interoperability in healthcare systems to facilitate access of mHealth apps to healthcare data in the cloud stored in multiple EHR/HIT systems. One of the main goals of FHIR is to represent the entities and procedures in healthcare as resources (FHIR

DSTU2, 2015). There are currently over one hundred forty-five resources (HL7 International, 2019) that can be represented using JSON, XML, or Turtle that can be utilized to map data from a healthcare system. Sample resources include: Patient, FamilyMemberHistory, Condition, Observation, Diagnostic Report, Medication, Immunization, AllergyIntolerance, AdverseEvent, etc.; and, for InsurancePlan, Coverage, EligibilityRequest, Claim, PaymentNotice, etc. The available resources can be accessed through the means of a RESTful API, which allows to connect healthcare interfaces with data sources that exist in the cloud. Different from SOAP (Simple Object Access Protocol) (W3C, 2007), which has been the dominant approach to manage web services interfaces over the past years and is utilized in HL7 v2, RESTful APIs are easier to understand and to implement as they rely on HTTP and Create, Read, Update, and Delete (CRUD) operations to develop services. In support of the integration of trust profiles and trust negotiation with FHIR, we have identified four security levels where security constraints are applied: the system level, which guards access to the entire EHR and encompasses all available resources; the resource type level, which guards access to all resources of a given type (e.g., Observation, Medication Statement, etc.); the resource level, which guards access to an individual instance of a resource identified by FHIR ID; and, the consent level, which allows the patient whose healthcare data is described by the resource to introduce security constraints in support of ONC's patient consent model (The Office of the National Coordinator for Health Information Technology, 2019).

One popular open-source library that implements the FHIR specification is the HAPI FHIR reference implementation (HAPI FHIR, 2014). HAPI FHIR was developed in the Java programming language and offers the features of FHIR in addition to other

features such as the ability to intercept the server (by using Java servlets (Oracle, 2013)) that processes the user's requests. HAPI FHIR offers the full FHIR REST API in Java and support for connecting HAPI FHIR to a back end database that stores the healthcare information. An instance of HAPI FHIR can be configured by extending the ResourceProvider interface and adding the resulting class to the instance's resource providers. The new ResourceProviders are written containing the functionality to interface with the local database by annotating methods utilizing annotations that mark which methods are invoked depending on the FHIR REST API calls. A controller may interface with a HAPI FHIR server by acting as a front end. A new URL endpoint following the FHIR REST standard is created to which a mobile device that supports trust profiles and trust negotiation connects. The controller engages in trust negotiation, then retrieves the requested FHIR data by forwarding the request to the HAPI FHIR installation if trust negotiation is successful.

2.4. The Connecticut Concussion Tracker (CT^2) App

The Connecticut Concussion Tracker (CT^2) app is a collaboration between the Departments of Physiology and Neurobiology, and Computer Science & Engineering at the University of Connecticut and Schools of Nursing and Medicine. The CT^2 app was created as the result of a new law passed in Connecticut (Connecticut General Assembly, 2015) requiring that concussions be tracked for kids between the ages of 7 to age 19 in public schools. The CT^2 app is used as proof of concept of the approach described throughout the remainder of the dissertation. Figure 2.2 shows a listing of the various screens in the CT^2 application, which will be described from left to right and top to bottom;

screens 1 to 4 in the first row and screens 5 to 8 in the second row. Screen 1 is the default login screen from which users can register an account or log in to an existing account. Screen 2 is the home screen and allows the user to create or retrieve a new entry for a student, or to view a list of open concussion cases. Screen 3 is the list screen and displays a list of students that the user has permission to read or edit. Screen 4 is the new student screen and allows the user to add general information about a new student such as: first and last name, the date of birth, the date of the incident, and gender. Screen 5 is the cause of injury screen that provides functionality to allow the user to provide details about the injury including: the sport being played, the location the injury occurred, the contact information, the location of the injury on the student, whether the student was wearing protective headgear, and any other additional details. Screen 6 is the symptoms within 48 hours screen that allows the input of symptoms the student has felt within 48 hours of the concussion incident. Inputs include: loss of consciousness and length of loss of consciousness, whether the parents were notified within 24 hours of the incident, whether the student was removed from the activity and who removed them from the activity, the concussion assessment tool used, and any other additional comments. Screen 7 is the injury follow-up screen and allows input of symptoms past the 48 hour period. Inputs include the timeframe in which symptoms were resolved, who diagnosed the concussion, whether there was any other post concussive symdrome diagnoses, the medical imaging used on the student, and other additional comments. The last screen, screen 8, provides for the input of when the student is allowed to return to the school. Inputs include days absent from school, schedule modification, whether a 504 plan is required, the date the student returned to the school, and the date the student returned to the sport.

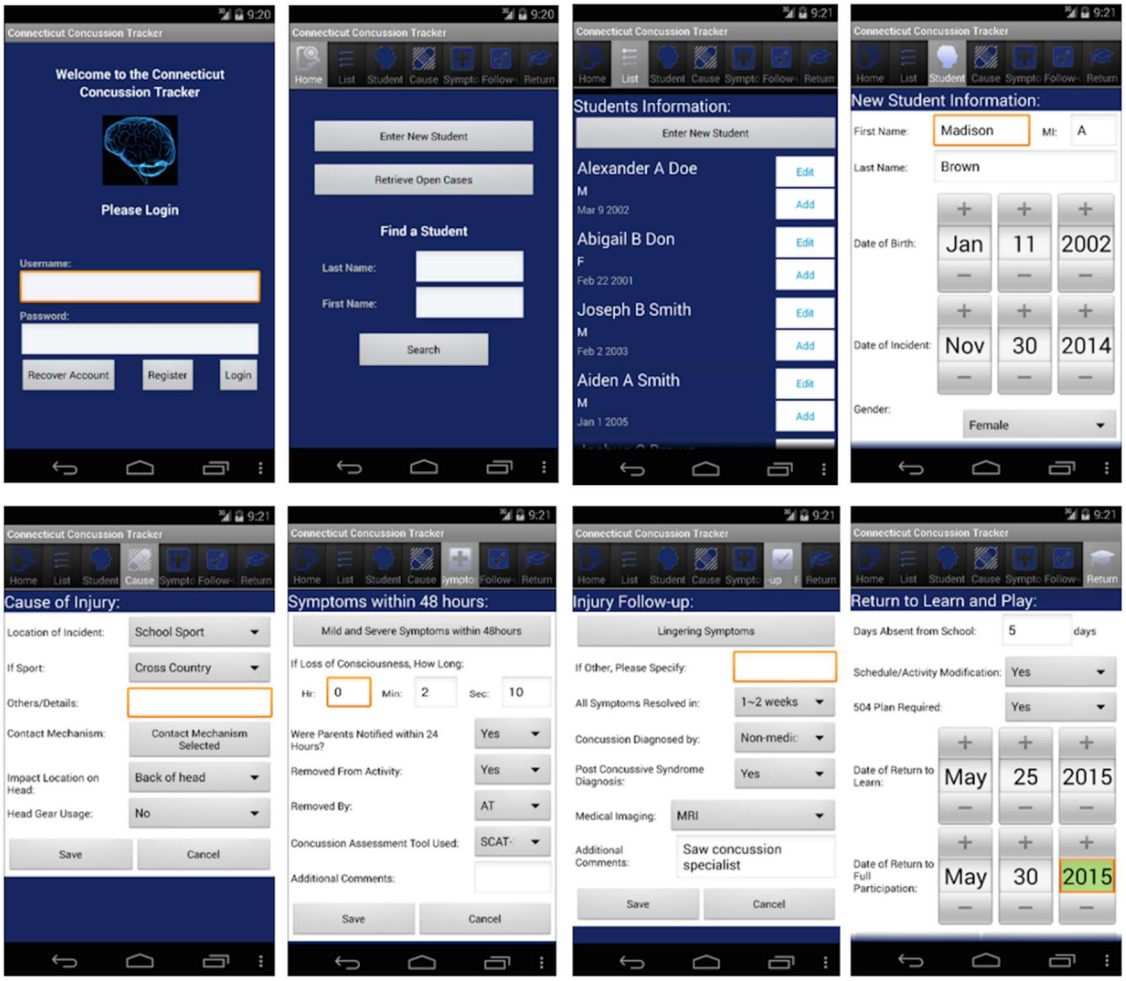

Figure 2.2. The CT^2 Application Screens.

Chapter 3
Infrastructure Requirements for Trust Negotiation

In this chapter, we describe the infrastructure requirements for trust negotiation and provide a detailed explanation of the different components involved in trust negotiation and the interactions among them with examples provided within the healthcare domain. This chapter consists of 4 sections. Section 3.1 describes Identity and Attribute certificates that form a verifiable container for the credentials that are necessary for communicating the components of a trust profile within a domain. Section 3.2 explains the process of trust negotiation and the communication process that occurs between two participants during the establishment of mutual trust through the utilization of a controller. Recall that to support the trust negotiation process, each participating HIT system (for the user or the data source), has a *controller* that facilitates the request from the user and utilizes the trust profile to determine if a user without a pre-existing relationship can access the requested data. Section 3.3 explorers the Trust Profile Certificate Infrastructure that adopts a chain-of-trust model that allows for the creation of new entries for a trust profile by establishing mutual trust between the trust profile entries of unknown healthcare providers and a method of verifying trust profile entries. Lastly, Section 3.4 reviews the structure of a controller's infrastructure that includes parts and operation in detail. All of the concepts discussed in Sections 3.1 to 3.4 support expected Contribution A: Infrastructure Requirements to Promote Trust Among Organizations Participating in Trust Negotiation. Section 3.3 also supports expected Contribution B: Trust Profile for Recording Complete Records of User

Access to Sensitive Data while Section 3.4 supports the implementation of expected Contribution C: Dynamically Generated Adaptive Access Control Policies. Finally note that the concepts throughout this chapter can be interpreted in a general manner to apply to any domain, but in order to make the discussion more relevant in support of the rest of the dissertation, we focus on the interactions of healthcare organizations. Concepts of Identity and Attribute certificates in Section 3.1, the trust negotiation and communication in Section 3.2, the trust profile certificate infrastructure in Section 3.3, and the controller infrastructure in Section 3.4 all are generalizable and applicable to other domains.

3.1. Identity and Attribute Certificates

A *certificate* is a file that provides a verifiable digital signature on the data contained within it. Certificates are widely used to verify that: the user has connected to the correct server, the connection is secure against man in the middle (MITM) attacks, and provides a secure manner to negotiate a symmetric key for efficient connection encryption. The most widely implemented standard for certificates, X.509 (Cooper, et al., 2008), provides both *identity certificates* and *attribute certificates* (Farrell & Housley, 2002). Although the X.509 identity certificate standard provides provisions for creating additional attributes, the attribute certificate is useful for shorter lived attributes and for decoupling the user's identification attributes from additional descriptive attributes. Pairing identity and attribute certificates together provides a platform for decentralized authorization management. The remainder of this section describes identity certificates and the chain of trust along with a detailed discussion of attribute certificates

An *identity certificate* is a file that provides ownership information by providing a verifiable public key. The certificate is used in a public key infrastructure to communicate

a public key that only the legitimate certificate holder can prove ownership of. By proving ownership of the public key, the owner also proves ownership of the certificate itself. When a certificate is created, the owner generates a public/private key pair. The private key is kept secret, while the public key is entered into a certificate signing request (CSR). When the certificate has been signed, the certificate acts as a notification as to the owner's public key and may be safely disseminated to any interested parties. When the owner of a certificate needs to be confirmed, it is sufficient to determine that the potential owner possesses the private key paired with the public key in the certificate, assuming that the private key has been kept secret by the owner. When another party needs to determine that they are communicating with the certificate's owner, a challenge is encrypted by the public key and sent to the potential owner. If the potential owner is in possession of the private key, the private key is used to decrypt the challenge. The owner signs the response with the private key and sends it back to the challenger. If the challenger verifies the response with the public key and the response is correct, the challenger knows they are communicating with the certificate's owner.

An identity certificate consists of multiple parts including: a version number, serial number, issuer name, validity period, subject name, subject public key, and a certificate signature. The *serial number* is an identifier of the certificate and must be unique for each certificate signed by an issuer. Therefore, each certificate may be uniquely identified by serial number and issuer. The issuer name describes the signer of the certificate. The *validity period* specifies the time period during which the certificate may be considered valid, consisting of not before and not after dates. The *subject name* specifies the owner of the certificate that possesses the private key and often consists of the host domain for which

the certificate was created. The *subject public key* is the public key of the owner generated as part of a public/private key pair and used for identifying the owner. The *certificate signature* is a digital signature of the certificate created by the certificate issuer at the time the certificate is created. The signature consists of a hash of the other information in the certificate signed with the certificate signer's private key. The certificate signer, known as a *certificate authority (CA)*, is the entity from the issuer field and is responsible for verifying the information in the CSR and creating the digital signature.

The digital signature provided by the CA on the identity certificate is required to bind the subject to the given public key to enable trust in the signed certificate. Without the signature to provide verification of the signature data, an attacker could simply provide a different subject public key paired with their own private key to hijack the connection. In order to ensure that the digital signature is valid, anyone using the certificate must also verify the certificate of the signing CA. The X.509 standard supports a *chain of trust*, or a linking of digital signatures from the "leaf certificate", or the first certificate in the chain, through intermediate CAs, to the *root authority*. The root authority is the last signature in the chain. The root authority's certificate is not signed by a higher CA, but is signed using the private key of the root authority itself, providing a *self-signed certificate.* The only method available for a root authority's certificate to be verified is if it is already present in the user's local *certificate store*, which is a local collection of certificates whose signatures are trusted automatically by the user. During verification, a user must obtain each certificate in the chain, and inspect the signatures starting from the leaf certificate to the root certificate until: one of the certificates matches a certificate in the certificate store or the chain ends without matching a certificate in the store. In the former case, the certificates

are verified and the leaf certificate is considered valid and trustworthy; in the latter case, the certificate cannot be verified and is considered untrustworthy. An example of the chain of trust is given in Figure 3.1. In the example, the trust flows from the Root CA Certificate to the Intermediate CA Certificate to the Leaf Certificate.

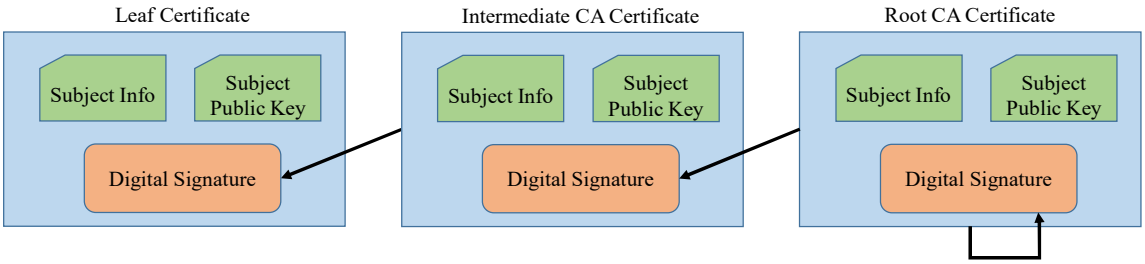

Figure 3.1. The Chain of Trust.

An *attribute certificate* (Farrell & Housley, 2002) is a digitally signed certificate that can act as an extension to an identity certificate, providing more detailed information about the owner of the identity certificate. Each attribute certificate can be paired with one identity certificate. Although attribute certificates are digitally signed, they lack the subject public key field that allows the certificate owner to authenticate themselves to the certificate and the data it holds. Attribute certificates are paired with an identity certificate through the identity certificate's serial number and issuer field. Although an attribute certificate can only be paired with a maximum of one identity certificate, multiple attribute certificates may be attached to the same identity certificate. Attribute certificates have fields similar to the identity certificate including: the version number, holder, issuer, signature, a serial number, and validity period. The body of an attribute certificate contains a list of attributes in key-value pair format. Like the digital signature of the identity certificate, the attribute certificate's signature is a computed hash of the other fields of the

certificate signed with the issuer's private key. An attribute certificate's authority is referred to as attribute authority (AA) and may be chained in the same manner as a certificate authority (CA). By utilizing attribute certificates in conjunction with identity certificates, the authentication and authorization data can be separated, with the identity certificate providing authentication for the certificate owner and the attribute certificate providing the data necessary for authorization. This allows the identity and authorization data to be long-lived while providing for the ability to expand upon or change authorization without the necessity of modifying the identity certificate.

3.2. Trust Negotiation Process

Trust negotiation (Winsborough, Seamons, & Jones, 2000) is the process of establishing mutual trust between two entities through the gradual exchange of credentials. In this context, trust is the ability for two entities to: believe in the authenticity of one another's credentials, utilize those credentials to determine the trustworthiness of each other with regards to handling sensitive data, and predict that each participant will handle any transferred sensitive data appropriately and securely. Each trust negotiation procedure begins with the requestor's request for sensitive data. The requestor contacts the controller over a secure connection and sends the controller the first set of credentials it has chosen to reveal. The controller receives the request for trust negotiation and retrieves a credential expression, represented by ψ, which takes the form of a logical expression that denotes the credentials that the requestor must possess to obtain access to the requested object. If the initial set of credentials is sufficient to satisfy the credential expression, the controller allows the requestor to access the requested object. Otherwise, the controller sends a notice

to the requestor indicating the remaining required credentials. The requestor may also request credentials from the controller if required. This allows the use of credentials that are sensitive by allowing the credentials themselves to be annotated with a desired trust level. Each round of trust negotiation results in a higher level of trust between the requestor and controller until the controller's credential expression is satisfied, or the requestor and controller no longer have any remaining credentials that may be sent. If the expression is satisfied, the request for trust negotiation is successful and the requestor obtains access to the requested object; otherwise, the request for trust negotiation has failed and the requestor does not obtain access. *Adaptive trust negotiation* (Ryutov, Zhou, Neuman, Leithead, & Seamons, 2005) refers to trust negotiation where the controller is able to adapt its credential expressions depending on the requestor's credentials and the requested object.

The usefulness and flexibility of the trust negotiation process has resulted in adaptations to mobile devices and healthcare (Vawdrey, Sundelin, Seamons, & Knutson, 2003). The credentials involved in trust negotiation are often encoded in certificates. Surrogate trust negotiation (Sundelin, July 2003) allows the use of a surrogate for mobile device trust negotiation to assist in the management and disclosure of credentials. A trusted server may act as a proxy to the mobile device, performing trust negotiation calculations on its behalf. Credential disclosure may be controlled by *trust agents*, which are autonomous programs that disclose secure credentials based on pre-defined credential access policies. A remote trust agent running on a surrogate must be completely trusted by the user because the trust agent manages both credential disclosure and the certificate's private keys.

As a healthcare focused example of adaptive trust negotiation, consider Physician Tom attempting to access his patient's health record at a remote hospital as shown in Figure 3.2. Physician Tom possesses credentials indicating his medical licensing, his current workplace, and that he is the patient's practicing physician (Certificates License, Employer, Inpatient in the upper left side of Figure 3.2 under Tom). Physician Tom uses his mobile device to connect to his local hospital's Local Trust Agent server in order to initiate a Trust Negotiation Request (line 1 in Figure 3.2) to the Remote Hospital's EHR. The EHR receives the Request for Patient Record (line 2 in Figure 3.2), creates a Credential Expression (line 3 in Figure 3.2), and sends the Credential Expression to Physician Tom (line 4 in Figure 3.2). The Credential Expression sent indicates that indicates that access to the record will be provided if: the requestor is licensed, employed at a known healthcare organization, and is currently treating the patient. The Local Trust Agent sends a credential with Tom's medical license and a request for a credential indicating that the EHR's security has been certified by an outside healthcare security consultant (line 5 in Figure 3.2). The EHR verifies the Medical License and sends a credential indicating the Security Certification back to the trust agent (line 6 in Figure 3.2). The Local Trust Agent, satisfied that the EHR is secure, sends the remaining current employment (Employer) and Inpatient treatment credentials (line 7 in Figure 3.2). The EHR receives them and verifies them. Now that the Credential Expression has been satisfied (line 8 in Figure 3.2), the EHR reports a successful trust negotiation attempt and sends the patient's Health Record to Physician Tom, who is now able to view the patient's health record on his mobile device (line 9 in Figure 3.2).

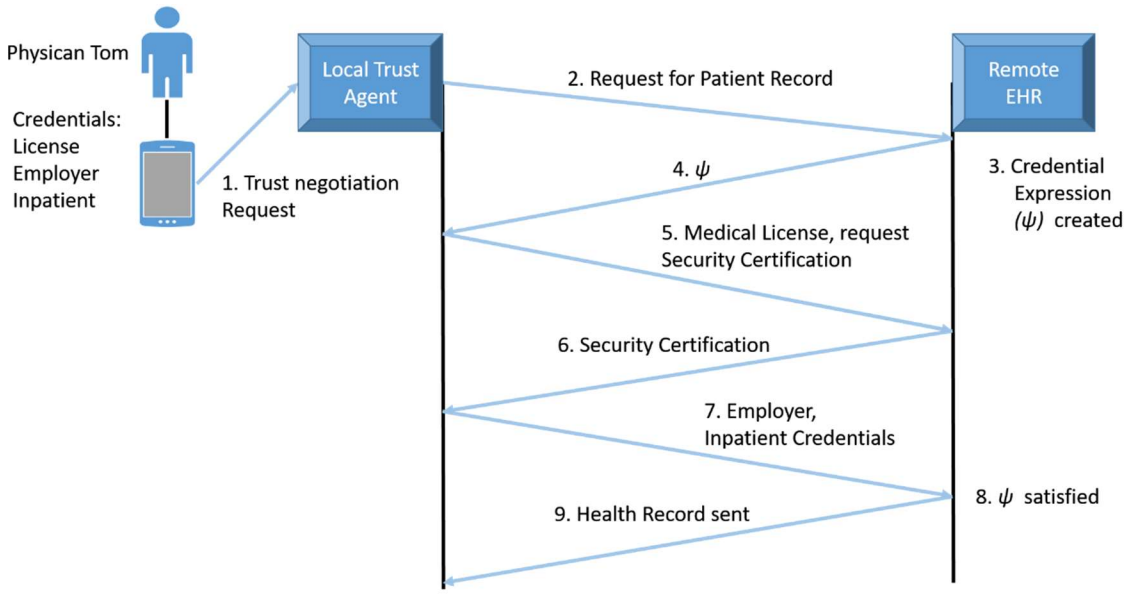

Figure 3.2. Trust Negotiation Request.

3.3. Trust Profile Certificate Infrastructure

The trust profile is capable of establishing trust between the requestor (e.g., physician, nurse, health insurance representative, etc.) and an organization's EHR by providing a standard set of credentials describing sensitive data access allowed by a peer organization (e.g., another hospital). Before the controller can make a decision as to whether the trust profile contains sufficient credentials to access the requested sensitive material, the credentials themselves must be valid to ensure that the credentials describe real accesses to other organizations. The trust profile's data can be trusted as valid if: the trust profile owner's ownership is confirmed through the identity certificate's public key, the digital signatures listed in the identity and attribute certificates are valid, and the controller confirms that the signers of the identity attribute certificates are authorized to create certificates for the trust profile entries. This third requirement ensures that a

controller only trusts the judgement of real organizations and prevents an attacker from creating and signing their own trust profile certificates.

In order to establish trust among the healthcare organizations participating in trust negotiation, they must each trust mutual medical authorities. A *medical authority* performs a function similar to the certificate authorities that sign certificates for secure communication for website login data. The medical authority is responsible for signing X.509 CA certificates for each healthcare organization's CA and AA. Medical authorities provide their root certificates for healthcare organizations to validate the digital signatures on the trust profile entries digitally signed by their peers, whose CA and AA certificates were signed by the medical authority. Unlike a certificate authority, whose role may be limited to verifying that the certificate owner of the certificate the CA signed owns the domain listed in the certificate, the medical authority is also responsible for ensuring that the healthcare organizations' private keys that match the public keys listed in their CA certificates are stored in a safe and secure manner. Theft of the private key threatens the validity of the portions of trust profiles signed with the stolen private key since those entries could be forged. A signature from a valid medical authority on a healthcare organization's CA and AA certificates builds trust by ensuring that proper security measures are being properly followed.

The establishment of trust from the medical authority, to the local healthcare organization, to the trust profile owner follows the X.509 chain of trust model as previously shown in Figure 3.1. In the chain of trust model, validation of a certificate starts with the last certificate in the chain, the leaf certificate depicted in the left side of Figure 3.1, and travels up the chain through one or more CA certificates until either one of the CA

certificates matches a certificate in the local certificate store or the last, self-signed root authority certificate is found. The medical authority, whose certificate is represented by the Root CA Certificate in the right side of Figure 3.1, establishes trust between the various healthcare organizations participating in trust negotiation by signing the CA and AA certificates of the healthcare organization in the middle of Figure 3.1 and making its own self-signed root certificate available. Any healthcare organization that places a copy of the medical authority's root certificate in their local certificate store indicates that any certificates signed by the medical authority are trustworthy. Using this process, those healthcare organizations also indicate that peer healthcare organizations' CAs and AAs that are signed by the medical authority are also trustworthy. This chain of trust flows downward into the certificates that comprise a healthcare professional's trust profile. Thus, trust flows from the medical authority to the CAs and AAs of healthcare organizations sharing sensitive data to the healthcare professionals employed by those healthcare organizations attempting to obtain sensitive data. Multiple medical authorities may be present on the network and healthcare organizations are free to choose which medical authorities will be trusted, as well as choosing individual healthcare organizations to trust directly through the local certificate store.

In the event of a security breach resulting in a problem with the private keys of a set of trust profile certificates, the CAs or AAs may issue an entry in their certificate revocation lists (CRLs) as specified by the X.509 standard (Cooper, et al., 2008). Likewise, the certificates of a compromised or misbehaving CA or AA may also be revoked by the medical authority if the CA or AA fails to meet proper security requirements or issues invalid trust profile entries. The X.509 standard provides for a CRL, which is a digitally

signed list of the serial numbers of certificates that should not be considered valid, and information regarding their current state of trustworthiness. During certificate validation, the certificate validator must consult an up-to-date CRL, with all trust profile entries whose certificates appear on the CRL not being considered acceptable for use in trust negotiation. Whether trust negotiation immediately fails depends on the listed reason for certificate invalidation. For instance, the CRL may list that hackers have stolen the owner's primary key, and thus proper ownership of the certificates cannot be ascertained. The CRL also provides the function of allowing employers to terminate trust profile entries indicating current employment should the employee change jobs.

Figure 3.3 depicts an example trust network with multiple medical authorities establishing trust among Family Medical Center (FMC), Saint Francis Hospital (SFH), and Hartford Hospital (HH). In Figure 3.3, the dashed arrows represent that the healthcare organizations' CA and AA certificates have been signed by that medical authority, and that the medical authority has performed security audits on the healthcare organizations' trust negotiation setup. The dashed arrows also represent that the local trust certificate store contains a copy of the medical authority's self-signed certificate, since the medical authority that signed the healthcare organization's CA and AA is trusted by default. The solid arrows represent trust placed in the medical authority by the local healthcare organization by placing the medical authority's self-signed certificate into the local trust certificate store. Each of these healthcare organizations is therefore able to trust the others' entries in trust profile certificates, since during certificate validation each certificate chain will result in eventually validating a medical authority certificate that appears in the local certificate store even though each healthcare organization has been audited and signed by

a different medical authority. This system allows for multiple competing medical authorities and prevents creating a single point of failure on the network, while still allowing trust to be created and disseminated with a distributed approach.

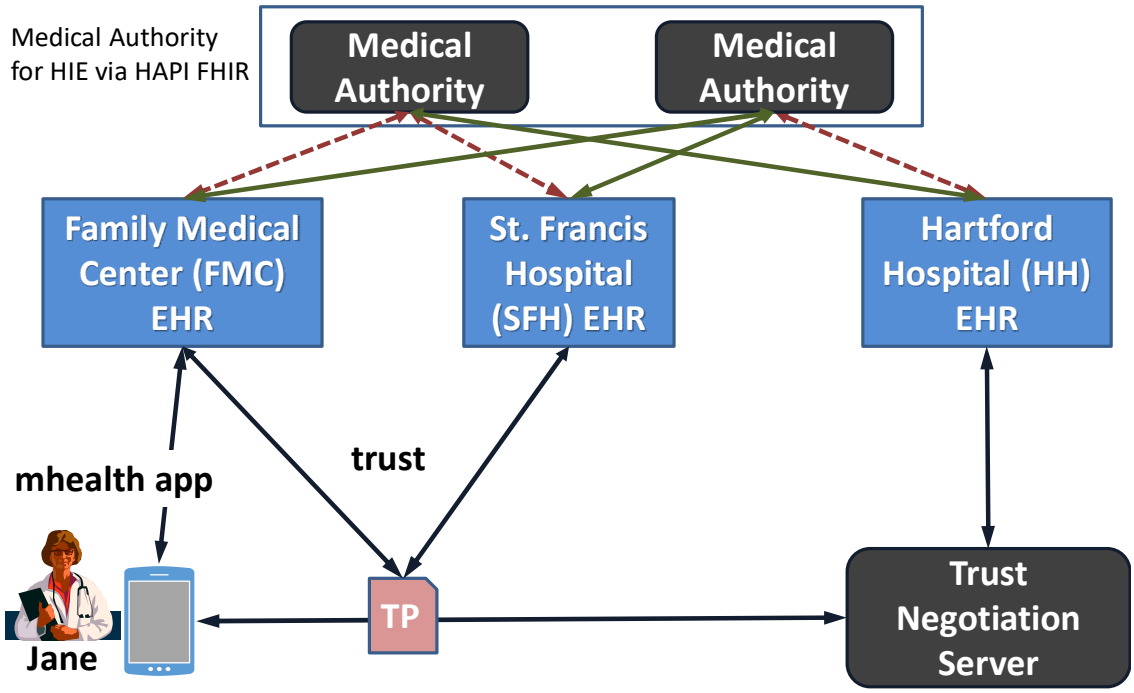

Figure 3.3. Example Network With Multiple Medical Authorities.

3.4. Controller Structure

The trust profile combined with an adaptive trust negotiation approach is dependent on the interactions occurring between the users' trust profile and the controllers. This adaptive approach and the organic growth of the trust profile requires a more complex controller. We have organized the Trust Negotiation Server into five distinct components as depicted in the boxed area of the right side of Figure 3.4. The five components are: the Trust Profile Validation component, the Security Policy Matching component, the Security

Policy Generation component, the Credential Generation component, and the Data Collection and Delivery component. The responsibilities of the five components include: verifying the trust profile's structure and content; determining proper ownership of the trust profile without needing to verify a specific identity; matching the user's chosen subset of the trust profile against a security administrator-defined security policy on the requested data; delivering the requested sensitive data from the EHR to the requestor, including any necessary modifications required by the adaptive approach specified in the security policy; and, communicating with the user and the local CA and AA to create a new entry for the user's trust profile. Finally note that the blue ovals in the Trust Negotiations server box of Figure 3.4 can be leveraged for other domains.

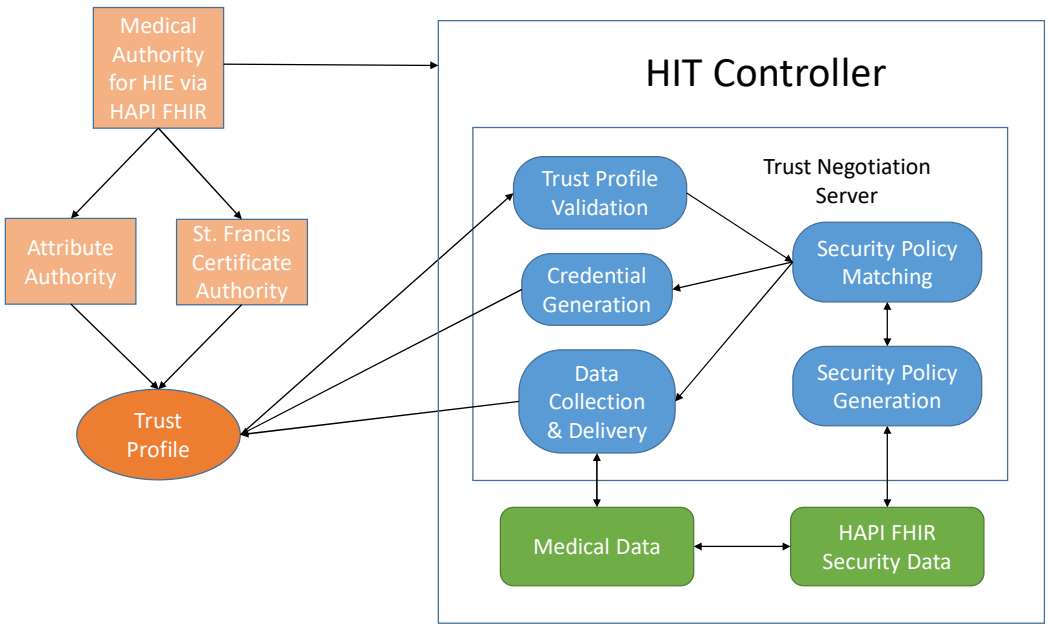

Figure 3.4. The Controller Structure.

When authorization through trust negotiation is requested by the user, the controller begins by receiving the request and validating the initial certificates passed by the user. The Trust Profile Validation component verifies the certificates so other components of the Controller are able to trust the credentials and determine if they are sufficient for data access. The Security Policy Generation component reads the security metadata attached to the requested Medical Data using the FHIR standard and uses this data as well as administrator provided Security Policy Generation rules to generate a credential expression that the Security Policy Matching component will match the user's Trust Profile against to determine whether access is granted. The Security Policy Generation component is also responsible for generating ancillary release actions, such as triggering logging records or redacting data that the user has not obtained enough trust to receive. The Security Policy Matching component is responsible for receiving the generated security policy as a credential expression and matching it to Trust Profile entries extracted from the previously validated trust profile certificates. This component also notifies the user if the credentials previously sent are insufficient and the nature of the credentials that must be presented in order to build further trust. Further presented Trust Profile credentials are passed to the Trust Profile Validation component to be verified before the Security Policy Matching component receives them for as many trust negotiation rounds as necessary. When the Security Policy Matching component decides the user has been successful in obtaining trust, it notifies the Credential Generation and Data Collection and Delivery components. The Credential Generation component communicates with the user to add trust profile entries detailing the successful access to data, while the Data Collection and Delivery component gathers the requested data, performs any necessary transformations on the data

(e.g., converting to a FHIR interoperability format, redacting data that should not be sent to the user, etc.), and performs any ancillary actions required by the security policy (e.g., dispatching emails to the hospital's auditor in the case of a high risk transaction).

Chapter 4
Trust Profile Model and Adaptive Trust Negotiation Approach

In this chapter, we describe our extensions to the adaptive trust negotiation framework introduced and discussed in Section 3.2 in order to provide a formal trust profile model that is able to represent all of the necessary concepts for tracking a user, a users trust profile, the controller, and all of the underlying concepts that are necessary to facilitate the trust negotiation process. In support of this, this chapter consists of 4 sections. Section 4.1 describes the trust profile's structure and usage in detail, focusing on all of the required constructs and the way that they interact with one another, that are required to support the formal model to be presented in Section 4.2. Section 4.2 presents and explains the formal model for supporting trust profiles in the trust profile negotiation process which allows each organization (e.g., a hospital or a medical practice) that supports trust negotiation to independently protect their health data while being able to endorse users as trustworthy from other organizations (other hospitals or medical practices or any HIT system) as well. Section 4.3 provides an example of the concepts presented in Sections 4.1 and 4.2 in the healthcare domain to clearly demonstrate the applicability of our work in practice in a realistic setting, fully illustrating the concept and content of a trust profile along with the trust negotiation that occurs with the controller to allow for a decision to be made about access to the requested sensitive data. Please note that our approach can work with any organization that requires control of highly sensitive information. Section 4.4 presents related work in applying trust negotiation techniques to the mobile and healthcare domains

and compares and contrasts the releated works to our research. All of the concepts discussed in Sections 4.1 to 4.3 support expected Contribution B: Trust Profile for Recording Complete Records of User Access to Sensitive Data while Section 4.3 also supports the implementation of expected Contribution C: Dynamically Generated Adaptive Access Control Policies.

4.1. The Trust Profile Concept

The trust profile is an extension to the concept of trust negotiation (Winsborough, Seamons, & Jones, 2000) that provides an automated form of trust gain over the course of a user interacting with a secure system over a period of time. This is accomplished by the accumulation of credentials over time as the user securely accesses the authenticated HIT systems to which they have been authorized. The result is a historical record that is summarized as the trust that the owner has obtained over the course of a long career handling sensitive information. As the user demonstrates the capacity to obtain, utilize, and protect this secure information, there is often an expansion in the user's responsibilities within an organization and an increased ability to access more heavily secured information. This trust is accumulated as the trust profile is updated over the lifetime that the user is working for any organization which requires access to sensitive data. The trust profile's strength is its ability to automatically model this increase in trust placed in the user without the need for constant human intervention. The trust profile records all of the owner's access to sensitive information, whether it is accessed locally or remotely; or accessed from the user's current employment with an organization (e.g., for our purposes a healthcare organization such as a hospital or a medical practice), past employment with an organization (e.g., a physician who previously worked as an intern or resident at a hospital),

or from another organization with no pre-existing relationship with the user. In the event that the user requires access to sensitive data in a time critical situation from another organization that the user has not been pre-authorized to access, the user can enter trust negotiation with the data's controller, utilizing a subset of individual records of sensitive data access within the trust profile as a set of credentials, supplemented by traditional static credentials common in trust negotiation such as medical licensing. Note that from this point on we omit the use of healthcare references as we discuss the concept of trust profiles in the formal model in more general terms.

In addition to the trust profile's ability to model the user's increase in trustworthiness by accumulating credentials throughout the user's career, we also introduce adaptiveness to the process by providing additional actions to the controller that may be undertaken depending on how the controller interprets the credentials offered in the trust profile and the details of the user's initial request for data resource access. Local administrators of the controller may define methods for the controller to fine-tune the level of trust present in a trust negotiation attempt by creating sets of release actions that the controller may undertake to increase security assurance while also providing methods to make sensitive data more available, particularly in time critical or emergency situations. For instance, an administrator may define that a requestor requesting access to a patient's drug list must present at least three credentials from the trust profile indicating that the requestor has successfully accessed patient drug data in the past and at least one credential from the trust profile indicating access to the health record of the requested patient from the controller in the past. The administrator creates a set of release actions indicating that access to the requested patient's health record may also be allowed if the access occurred

at another organization. In the case that the access occurred at another organization, access would still be allowed from the requested organization, whose controller would note the access in the audit log as a higher risk transaction and a notification would be dispatched to the organization's local auditor. This allows the requestor multiple secure paths to access sensitive, time-critical data without compromising health record security and for swift responses on the part of the local organization should sensitive health data be obtained inappropriately.

Every stakeholder (e.g., physicians, nurses, specialists, pharmacists, insurance agents, healthcare researchers, therapists, etc.) must be provided with a trust profile to access secure data when they first have a need to access sensitive data. Note this might also occur as part of the overall authorization and authentication process when new employees are created and given permissions on the local database. The stakeholder's employer is responsible for initializing the trust profile, which marks the stakeholder as having been manually vetted by the organization during pre-employment and having been found to be trustworthy in handling highly sensitive data. If the stakeholder later leaves the employer and finds employment elsewhere, the stakeholder retains the trust profile and the data contained within, as the trust profile is a permanent record of the stakeholder's sensitive data access over an entire career. These are essentially credentials that an individual takes with them as they move between different locations to administer care to patients. The stakeholder no longer has entries in the trust profile indicating that the owner is currently an associate of the previous employer, but retains data indicating previous employment during the period when the entries were made. The stakeholder is then free to obtain employment with a new organization, obtains a new entry indicating employment with the

new organization, and may use any previous entries in the trust profile during trust negotiation.

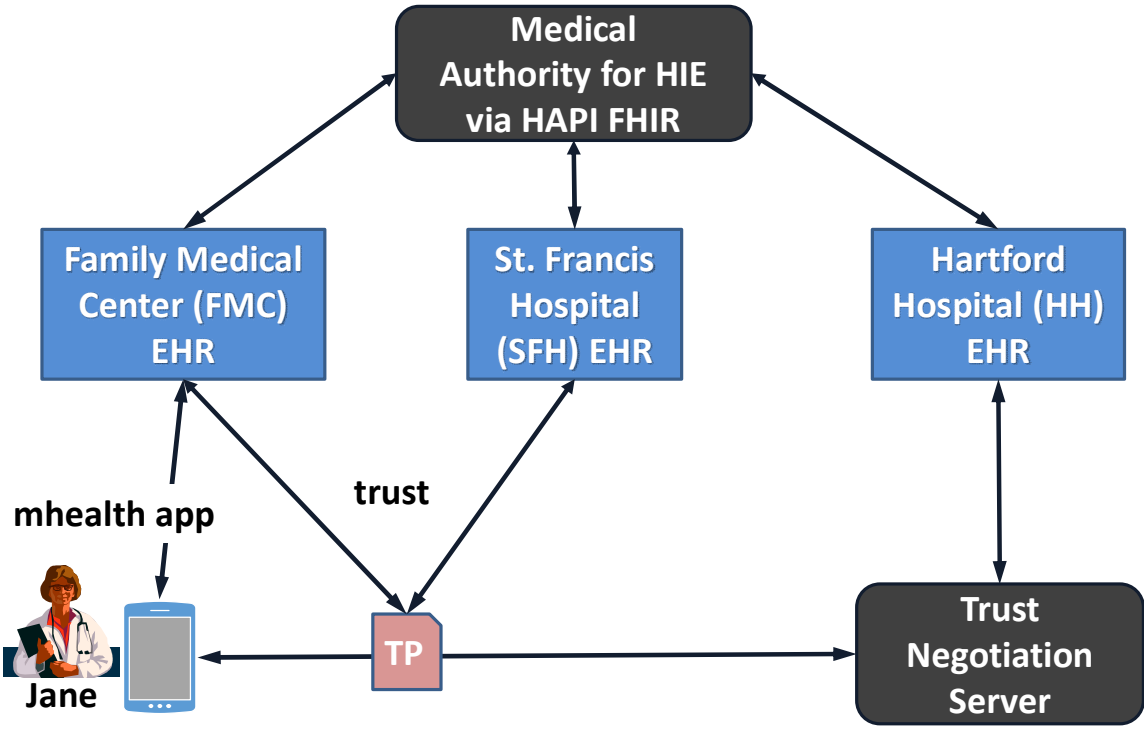

Figure 4.1. Example Trust Profile Negotiation.

Figure 4.1 illustrates the connection between the trust profile owner, the owner's employers, and the controllers. In Figure 4.1, Jane, a medical provider, possesses a mobile device from Family Medical Center (FMC) loaded with their mHealth app. The mHealth app is capable of accessing FMC's EHR with traditional credentials, but also supports requesting access from other EHRs, such as St. Francis Hospital (SFH) and Hartford Hospital (HH) through trust profile-based trust negotiation. The Trust arrows to/from FMC and SFH to the Trust Profile (TP) indicate that Jane has entries in her Trust Profile from both FMC and SFH, meaning that Jane has previously obtained access to sensitive data

from FMC and SFH and that those accesses have been recorded in the Trust Profile. Should Jane require access to her patient's health data located at HH, the mHealth app will send a subset of the Trust Profile, compiled into a digital wallet, to HH's Trust Negotiation Server. The entries chosen from the Trust Profile indicate the level of trust from both FMC, Jane's current employer, and SFH, a different healthcare organization Jane established trust with previously utilizing trust negotiation. HH's Trust Negotiation Server finds indications in the Trust Profile from FMC indicating that Jane is a current employee and has been manually vetted, while the entries from SFH indicate additional trust. HH's Trust Negotiation Server decides whether the credentials are sufficient, and if negotiation successful returns the requested data and a new entry for the Trust Profile describing the access.

The trust profile is implemented as a collection of identity and attribute certificates. Attribute certificates encode the trust profile data in a key-value pair relationship. The identity certificate serves as a method of authenticating ownership of the attribute certificates presented during trust negotiation to the user. The dual use of identity and attribute certificates is required because the X.509 standard does not provide a method of verifying ownership of an attribute certificate using public key infrastructure. Instead, each attribute certificate is associated with an X.509 identity certificate that does provide ownership verification through the listed public key. During trust negotiation, all attribute certificates sent must also be paired with their identity certificate or ownership cannot be verified. Once ownership of the identity certificate has been verified by verifying that the requestor possesses the private key associated with the listed public key in the identity certificate, the attribute certificates' ownership is verified by affirming their association to

a previously verified identity certificate. The encoded identity information stored within the identity certificates is not required to specifically identify the trust profile owner, but to ensure that the requestor is the owner of the attribute certificates chosen to form a subset of the trust profile for trust negotiation. The user obtains one identity certificate from the controller of each system accessed during the first successful attempt to request sensitive data. The identity certificate does not contain any access history specific data.

The attribute certificates contain individual entries that describe an access. Each sensitive data access results in at least one new attribute certificate, which is associated with the identity certificate obtained from the controller. Thus, a trust profile is comprised of one or more identity certificates, each representing at least one access from a controller; and each identity certificate will have one or more attribute certificates attached to it, each attribute certificate representing one aspect of access to one specific requested resource. These attribute certificates may represent: access to a requested object, the sensitivity level of the requested object, or a sensitivity level representing the highest sensitivity object the trust profile owner has obtained from the controller of that system. A fourth attribute certificate is provided that represents a current employer of the owner and denotes that the owner has been manually vetted by the employer during the course of employment. This fourth attribute certificate must be assigned manually by the employer. This combination of identity and attribute certificates provides the flexibility of being able to encode multiple accesses to sensitive data to the same healthcare organization using attribute certificates, while also minimizing the number of ownership verifications of identity certificates. Minimizing the number of public key ownership verifications is important because the calculations necessary to verify a public key are relatively slow and difficult to perform,

which may overtax less powerful CPUs on mobile devices or trust negotiation servers handling multiple concurrent requests.

An example trust profile subset is displayed in Figure 4.2. The physician illustrated has obtained secure access to Family Health Center, UCHC, and SFH in the past and possesses an identity certificate from each, representing the trust he has obtained from each organization. On the first access to each of these organizations, the controller has informed him of successful access and requested a new public key for the identity certificate. The attribute certificates associated with the identity certificate, whose associations are represented with arrows connecting the certificates, represent aspects of each successful access to the controller that created the associated identity certificate. During a new request for sensitive data, the physician may choose any combination of attribute certificates and their associated identity certificates and present the accesses the certificates represent as proof of trust obtained from those systems. Further accesses to systems that issue the identity certificates result in new attribute certificates detailing the access with new associations to that identity certificate, while access to the previously unknown Hartford Hospital as pictured will require the generation of a new X.509 identity certificate and new attribute certificates for the specific data accessed.

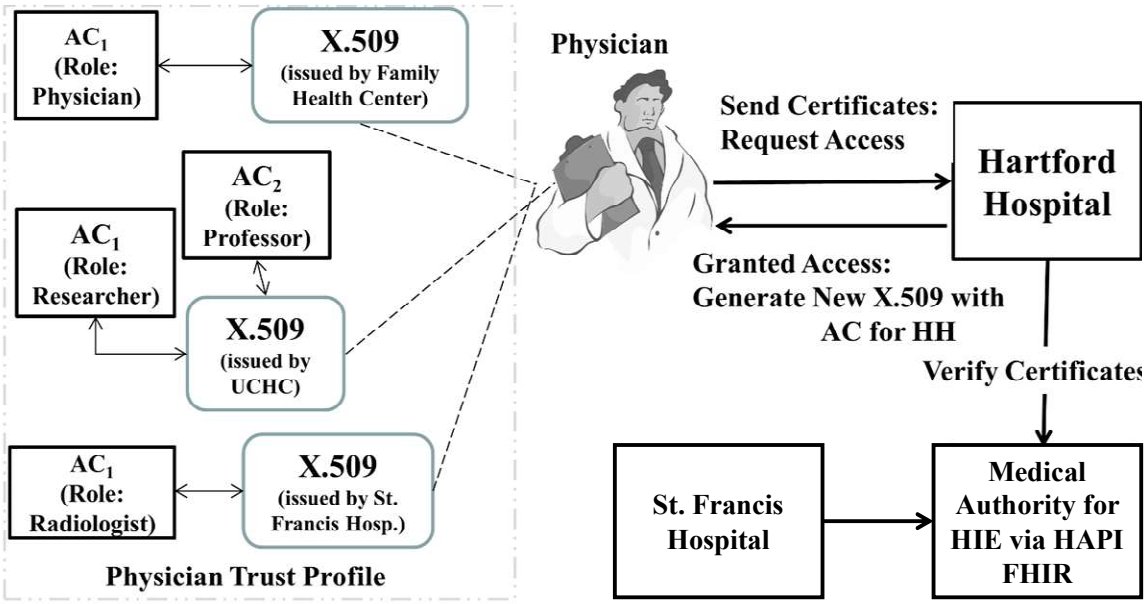

Figure 4.2. Example Trust Profile Structure.

4.2. A Trust Profiling Model for Trust Negotiation

In this section, we propose and explain a model for trust negotiation that presents a new trust profile that utilizes role-based access control (RBAC) (Ferraiolo, Sandhu, Gavrila, Kuhn, & Chandramou, 2001), mandatory access control (MAC) (Bell & La Padula, 1976), and attribute-based access control (ABAC) (Hu, et al., 2014) to model the sensitivity of information being requested, governed by role with credentials captured in attributes via ABAC. Recall that *trust* is the ability for two entities to believe one another and *trust negotiation* (Winsborough, Seamons, & Jones, 2000) is a process of two entities with no pre-existing knowledge of one another building trust by exchanging digital credentials. Our work specifically extends the trust negotiation approach by creating a living trust profile containing a complete record of the user's access history. The remainder of this section presents a formal model definition for the trust profile that is characterized

by the confidentiality of requested information to be accessed, role-based capabilities, and attributed-based access control to capture credentials. Note that this model is independent of the healthcare domain, it is a realization of the trust profile concepts as given in Figures 1.1 and 1.2. Continuing on with the discussion in the early portions of this dissertation, we leverage the healthcare domain and associated examples to explain the model concepts.

To begin the discussion of the formal model for trust profiles, we define a number of key terms.

Defn. 1: A *User,* $\mathcal{U} = \langle u_{id}, u_{name} \rangle$, is a trust profile owner identified by id and name.

Defn. 2: A *Requestor,* $\mathcal{R} = \langle r_{id}, r_{name} \rangle$, is a user engaged in trust negotiation by id and name.

Defn. 3: A *Controller,* $\mathcal{C} = \langle c_{id}, c_{name} \rangle$, accepts requests for trust negotiation by id and name.

Defn. 4: A *Security Agent,* $\mathcal{SA} = \langle sa_{id}, sa_{name} \rangle$, is an autonomous entity that manages receiving and disclosing credentials on behalf of a *Requestor* or *Controller* denoted respectively as $\mathcal{SA}_{Requestor}$ or $\mathcal{SA}_{Controller}$ identified by id and name.

Defn. 5: A *Credential Expression,* ψ, is a logical expression that represents the credentials necessary to establish trust between a *Requestor* ($\mathcal{SA}_{\mathcal{R}_{equestor}}$) and *Controller* ($\mathcal{SA}_{\mathcal{C}_{ontroller}}$).

Defn. 6: A data resource is defined in two different steps;

a. A *Data Resource Descriptor,* $\mathcal{DR}_{\mathcal{D}_{escriptor}} = \langle \mathcal{DR}_{\mathcal{T}_{ype}}, \mathcal{DR}_{\mathcal{CRID}} \rangle$, provides a $\mathcal{DR}_{\mathcal{T}_{ype}}$ that identifies the type of resource requested (e.g., a FHIR patient

resource , an object in JSON or XML format) and $\mathcal{DR}_{CRUD}$ a set of CRUD operations (*CREATE* | *READ* | *UPDATE* | *DELETE*) *R* wishes to perform on the requested object.

b. A *Data Resource,* $\mathcal{DR} = <\mathcal{DR}_{Descriptor}, \mathcal{DR}_{ID}>$, is identified by a $\mathcal{DR}_{ID}$ unique identifier and $\mathcal{DR}_{Descriptor}$ the type of resource requested. The $\mathcal{DR}$ represents a portion of information of interest.

Defn. 7: A *System,* $\mathcal{S} = <\mathcal{S}_{Name}, \mathcal{S}_{ID}, \mathcal{DRS}_{set}>$ is identified by $\mathcal{S}_{ID}$ unique identifier, $\mathcal{S}_{Name}$ the name of the system (e.g., an HIT system such as an EHR), and $\mathcal{DRS}_{set}$ the set of all data resources (e.g., the set of all FHIR resources in an EHR) for the system as given in Defn. 6.

Defn. 8: A *Role, r,* is a designation that represents the types of tasks the role's owner is expected to perform (e.g., physician, psychiatrist, nurse, etc.).

Briefly, we review the definitions. *Controllers* guard data and determine access to a data resource $\mathcal{DR}$ whereas *requestors* initiate the request for a resource. A credential expression, denoted ψ, represents the credentials that must be presented by the requestor to the controller to release a resource. If a set of credentials c satisfies ψ, it is denoted as $sat(c, \psi)$ (see Defn. 5). In healthcare, a participant would be a physician attempting to access an EHR or a hospital that possesses a patient's EHR. A controller would be a repository of public health data or a hospital that possesses a patient's EHR, and credential expressions would be utilized to determine the records of access to health records that must be present in the physician's presented trust profile to obtain access to a new health record. During negotiation, the requestor and controller alternate exchanging credentials through

security agents $\mathcal{SA}$ (see Defn. 4). The requestor and controller each possess their own $\mathcal{SA}$ that is responsible for releasing the owner's credentials and receiving the other participant's credentials. The process of exchanging credentials, depending on the level of trust established, continues until either the requestor has established enough trust with the controller to complete the request, or it is determined that trust cannot be established. If the exchanged credentials are insufficient to establish trust, a *Server Governance Policy (SGP)* is exchanged to alert the other participant to the credentials required to complete the negotiation. Defn. 6a and 6b provide methods for identifying specific records that $\mathcal{R}$ is performing the negotiation to obtain. The $\mathcal{DR}_{\mathcal{D}escriptor}$ is comprised of a $\mathcal{DR}_{\mathcal{T}ype}$, which denotes the type of FHIR resource requested, such as a patient medication, and a $\mathcal{DR}_{\mathcal{CRUD}}$ which indicates the CRUD operations the requester $\mathcal{R}$ wants to perform on the requested resource.

The user $\mathcal{U}$ corresponds with the Physician pictured in Figure 4.2 When the Physician sends a request for trust negotiation, the Physician becomes the Requestor $\mathcal{R}$. The u_{id} is the unique combination of the issuer and serial number present on the identity certificates in the trust profile while u_{name} is specified in the domain field as a name@domain pairing. These identity certificates are represented in Figure 4.2 in the X.509 identity certificates provided by Family Health Center, UCHC, and St. Francis Hospital. The controller as presented in the formal definition refers to the HIT Controller structure presented in Figure 3.4 introduced in Section 3.4. The $<c_{id}, c_{name}>$ pairing represents the unique serial number/issuer combination listed on the HIT Controller's Medical Authority certificate and the name is the controller's domain name. The controller is responsible for protecting the data of the system in Defn. 7, which is represented by the Medical Data in Figure 3.4. The *DR* is a request for the Medical Data, whose data is

categorized by type and annotated with HAPI FHIR Security Data utilized by the HIT Controller to generate the credential expression ψ. The role specified in Defn. 8 refers not only to the role the user *U* has chosen to perform trust negotiation under, but also references the trust profile entries that record the user's role at the time of a sensitive data access during a successful trust negotiation. These records are illustrated in the Trust Profile presented in Figure 4.3.

Given the initial set of definitions, the next set of definitions formalizes the components and structure of the trust profile including: access history properties, digital signatures, access history records, identity certificates, attribute certificates, the trust profile itself, and the digital wallet.

Defn. 9: An *Access History Property*, $\mathcal{AHP} = <p_{type}, p_{value}>$, is a single property that describes one attribute describing past access to a resource.

Defn. 10: An *Issuer,* $\mathcal{I} = <id_{Issuer}, PubKey_{Issuer}>$, is a controller that creates entries for a requestor's trust profile.

Defn. 11: A *Digital Signature,* $\mathcal{DS} = sign(PrivKey_{Issuer}, hash(IC))$, is a cryptographically-signed assurance that the signed content is both valid and unaltered since the information has been signed. A *DS* establishes the authenticity of a portion of the trust profile. The signature is created by encrypting a hash of the content the user's $\mathcal{IC}$ or $\mathcal{AC}$ with the issuer's private key.

Defn. 12: An *Access History Record,* $\mathcal{AHR} = < \mathcal{AHR}_1, \mathcal{AHR}_2, \ldots, \mathcal{AHR}_n>$, is a set of properties describing access to one resource. The number and types of properties listed in an $\mathcal{AHR}$ vary depending on the resource whose access they describe.

Defn. 13: An *Identity Certificate,* $\mathcal{IC} = <id_{User}, PubKey_{User}, id_{Issuer}, ds_{Issuer}>$, binds all attribute certificates issued for a single user from a single organization to a public key, allowing controllers to determine ownership of the attribute certificates by asking the requestor to prove knowledge of the associated private key (a cryptographic challenge).

Defn. 14: An *Attribute Certificate,* $\mathcal{AC} = <ic, ds_{issuer}, type, t, opt>$ contains information to be tracked on a requestor's actions over time and is a five tuple that has: an identity certificate, *ic*, per Defn. 13; a digital signature, *ds*, per Defn. 11; the *type* of the $\mathcal{AC}$; a timestamp indicating the time of access; and, a set of optional, *opt*, values depending on type. The four types of $\mathcal{AC}$s with opt information are:

a. $\mathcal{AC}_{DataResourceAccess}$: A record of a single access to a specific resource in a particular system by role $(\mathcal{AKR})$ where $opt=<u_{id}, r_{id}, dr_{id}, sys_{id}>$ are the user, assigned role, data resource, and system.

b. $\mathcal{AC}_{Affiliation}$: An affiliation of an organization on a network where $opt=<u_{id}, affil_{id}>$ are user and affiliation ids for a given user.

c. $\mathcal{AC}_{DataResourceConfidentiality}$: A confidentiality level for a specific resource where $opt=<u_{id}, cl, dr_{id}, sys_{id}>$ for a given user is the clearance level of the data resource and its system.

d. $\mathcal{AC}_{SystemConfidentiality}$: A confidentiality level that is the highest level of sensitivity for each system that has been accessed by the user at any time where $opt=<u_{id}, cl, sys_{id}>$ for a given user is the clearance level of the system.

Note that in our model, $\mathcal{IC}$s and $\mathcal{AC}$s are created upon successful trust negotiation by a controller for future use of the requestor (the user). Consequently, the exact properties used to describe the access are dependent on the controller and the domain. For instance, when accessing electronic healthcare records (EHRs) in the healthcare domain, the properties may include: the role under which access is allowed, the exact record that was accessed, the type of record accessed (e.g., MRI scan, general health record, drug list, etc.), the patient whose record was accessed, whether the requestor was an employee or the access was remote at the time of access, and a timestamp denoting the exact time of access.

Affiliation is an endorsement of a subset of the user's trust profile by a controller that has vetted the user more thoroughly and endorses the user's continued use of trust negotiation to obtain access to related resources. For instance, in the healthcare field, a physician participating in trust negotiation would have current affiliations with his/her current employer(s), who would be a healthcare organization trusted by other controllers within the healthcare network. A controller that grants a user an affiliation serves the dual purpose of providing initial entries into the user's trust profile, establishing assurance that the user is in good standing, and providing additional assurance that the user has a need-to-know when the controller protects need-to-know data. Note that affiliation exists in both an $\mathcal{AC}_{DataResourceAccess}$ as a property and at a higher level in the trust profile (see Defn. 15). Affiliation attached to a subset of the trust profile indicates *current affiliation*, or whether the requestor is currently affiliated with the controller noted in the trust profile, while the affiliation property in an $\mathcal{AC}_{DataResourceAccess}$ denotes whether the requestor was affiliated with the controller *at the time of access*. This is useful for situations when the affiliation was with a past employer and the user is employed elsewhere.

During a request for sensitive information, the context in which the request is made is important. Different requestors may have a legitimate need to request the same resource, but as access history is highly customized, the evaluation of trust profiles must accommodate a wide variety of entries. This requires the controller to be flexible in accepting the credential expressions generated for each attempted negotiation. For example, a family physician requesting a patient's electronic health record from a specialist (e.g., cardiologist) the patient has seen recently can be reasonably expected to have entries in his/her personal trust profile showing a history of access to the patient's records in the local EHR. However, a physician working in an ER is much less likely to be treating a patient he/she has seen before, but should be able to present a trust profile indicating the treatment of multiple patients in an ER setting. In support of this functionality, the trust profile model has definitions for a data resource $\mathcal{DR} = < \mathcal{DR}_{Descriptor}, \mathcal{DR}_{ID}>$ with an associated type $\mathcal{DR}_{Type}$ from Defns. 6a and 6b coupled with a request context. The next definition pulls all of the concepts together to define a trust profile:

Defn. 15: A *Trust Profile,* $\mathcal{TP} = <tp_{name}, u_{id}, tp_{AS}, tp_{DRAS}, tp_{DRCS}, tp_{SCS}, tp_{DW} >$ where:

- tp_{name} is the name to identify the profile,
- u_{id} is the unique identity of a user,
- tp_{AS} is the Affiliation Set (AS) of $\mathcal{AC}$s $\mathcal{AC}_{Affiliation}$ from Defn. 14b that contains all of the user's employment(s) or direct contact with the issuing organization,
- tp_{DRAS} is the Data Resource Access Set (DRAS) that contains a set of $\mathcal{AC}_{DataResourceAccess}$ from Defn. 14a where each $\mathcal{AC}_{DataResourceAccess}$ represents access by role to a data resource,

- tp_{DRCS} is the Data Resource Confidentiality Set (DRCS) that contains a set of $\mathcal{AC}$s $\mathcal{AC}_{DataResourceConfidentiality}$ from Defn. 14c where each $\mathcal{AC}_{DataResourceConfidentiality}$ represents the confidentiality of an accessed data resource,
- tp_{SCS} is the System Confidentiality Set (SCS) that contains a set of $\mathcal{AC}_{SystemConfidentiality}$ from Defn. 14d where each $\mathcal{AC}_{SystemConfidentiality}$ represents the confidentiality of a specific accessed system, and
- tp_{DW} is the Digital Wallet that contains a set of attribute certificates that are a subset of $\mathcal{AC}$s from tp_{AS}, tp_{DRAS}, tp_{DRCS}, and tp_{SCS} that represents the credentials being submitted by the requestor.

In support of the process, we conceptualize the Digital Wallet as the set of subject attributes the requestor ($\mathcal{SA}_{Requestor}$) sends to be evaluated. The security agents $\mathcal{SA}_{Requestor}$ and $\mathcal{SA}_{Controller}$ manage the disclosure of credentials for the duration of the negotiation. As illustrated in Figure 4.3, the Trust Profile is gradually released from $\mathcal{SA}_{Requestor}$ to $\mathcal{SA}_{Controller}$. The Trust Negotiation component receives the request for a resource and the Trust Profile from $\mathcal{SA}_{Controller}$. The resource is annotated with attributes of its own, represented by *Resource Object Attributes* in Figure 4.3, that include the Patient's ID, the physician assigned to the patient as the Primary Physician, the patient's Appointment Time, and the Sensitivity Level of the resource. When the Trust Negotiation component has received the credentials from the Trust Profile, the PDP attempts to match the subject attributes in the Trust Profile to the Resource Object Attributes and the Environment Conditions through ψ, which is created by the Trust Negotiation component from the Access Control Rules based on the type of resource being requested, the requestor's Role, and the Sensitivity Level of the

resource from the Access Control Rules. The Sensitivity Level of the requestor is determined based on the type of resource being requested, the Role of the requestor, and the contents of the Trust Profile.

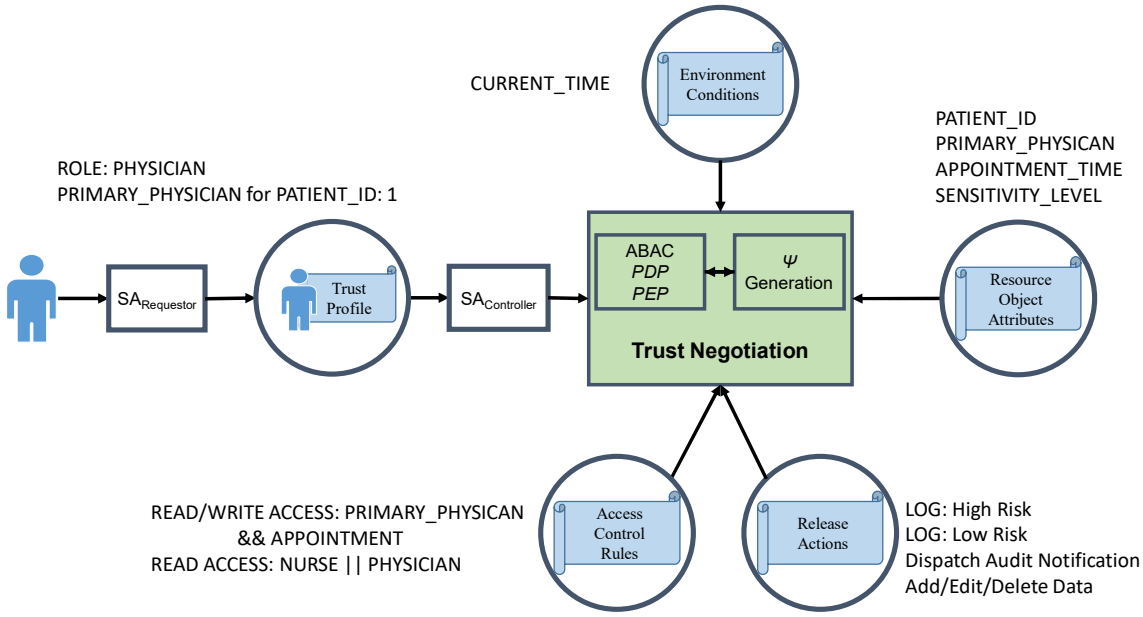

Figure 4.3. Integrated Trust Profile, ABAC, and Trust Negotiation.

For example, suppose a DR is an EHR that contains mental health data tagged with a sensitivity level of V (see Table 2.1), indicating that the mental health data is considered very sensitive and must be treated with utmost care. The requestor is the patient's psychiatrist and has multiple entries indicating access to the patient's health records. The psychiatrist initiates a request for the EHR under the psychiatrist role. The credential expression, ψ, generated by the Trust Negotiation component, will allow access if the trust profile indicates previous treatment of the patient and the psychiatrist meets the required V sensitivity level. The Trust Negotiation component determines that the sensitivity level will be met if the psychiatrist presents a trust profile indicating that he/she has accessed

data under the psychiatrist role, has accessed the patient's data specifically, and the patient is known to currently have an appointment with him/her. The psychiatrist's trust profile fulfills the first two requirements, and the PDP indicates that the current time from the Environment Conditions matches an appointment noted in the patient's records. The psychiatrist meets the sensitivity requirement, a new entry is made in the trust profile indicating successful access to the requested data, and the requested data is sent to the psychiatrist. The presented trust profile has fulfilled the requirements and the release actions LOG: High Risk and Dispatch Audit Notification are executed for the sensitivity level V transaction to provide further assurances of resource security, stored as part of AHR.

The final set of definitions is for the interactions between the trust profile, the request context, and the credential expression during trust negotiation.

Defn. 16: A *Request Context,* $\mathcal{RC}$ is a three-tuple that combines the resource, trust profile, and role as: $\mathcal{RC} = < DR, tp_{DW}, r >$

Defn. 17: The controller's credential expression $\mathcal{C}_\psi$ is a logical expression whose clauses correspond to credentials c that must be present in $\mathcal{TP}$ to allow access to $\mathcal{DR}$.

Defn. 18: The controller's set of release actions, $\mathcal{RA}$, is defined as a set of actions for the controller to execute depending on the received credentials if the credential expression ψ is satisfied.

Defn. 19: The credential expression ψ is satisfied iff the presented credentials have established sufficient trust to release the resource.

Defn. 20: A *Transaction,* $\mathcal{T}$, is a series of data exchanges between *Requestor* and *Controller*, initiated by the *Requestor* with an $\mathcal{RC}$, continues with disclosures of tp_{DW}, and ends when the *Controller* determines trust cannot be established or when ψ is satisfied. If ψ is satisfied the *Requestor* receives a new entry in the trust profile, receives the requested data, and the *Controller* executes all actions in $\mathcal{RA}$.

Defn. 21: The *Transaction,* $\mathcal{T}$, is valid iff ψ is satisfied and the controller has executed actions specified in $\mathcal{RA}$. $valid(T) = sat(c, \psi) + \mathcal{RA}$

When a request for trust negotiation is initiated, the requestor creates a *Request Context* (Defn. 16), which consists of the resource (e.g., patient EHR, public health data, etc.) the requestor would like to access, initial trust profile data, and the role the requestor possesses in his/her trust profile and would like to use for the trust negotiation. Using the request context, the controller creates a credential expression that represents the constraints that the trust profile must satisfy to obtain access to the requested resource. The controller may also take other actions depending on which constraints are satisfied. For instance, a family physician seeing a new patient may not have an access history in the trust profile indicating past treatment. A trust profile that has an affiliation with an organization but no previous interaction with the patient may result in a decision to allow access and send a notification to the controller's organization's auditor that the transaction requires review.

During the trust negotiation procedure, a subset of the trust profile chosen by the requestor acts as a set of credentials whose release is negotiated during trust negotiation. Each record of access in the access history forms one credential. Once a base level of trust is established between the controller and the requestor, the requestor chooses to send

subsets of the trust profile to satisfy the remaining portions of the controller's credential expression to obtain access to data. The controller's credential expression is generated subject to the data being requested and the role the requestor possesses for the session. If the requestor is able to satisfy the controller's credential expression and access is obtained, the controller adds its own entries to the user's trust profile that can satisfy the requirements of future attempts at trust negotiation with any other controller. Additionally, the controller may take other actions depending on the credentials presented, such as calculating transformations (addition, modification, deletion, etc.) on the data or dispatching audit notifications. Figure 4.3 illustrates a modification of the ABAC model from Figure 2.1 that incorporates the trust profile and trust negotiation. The credential expression, ψ, shown in Figure 4.3 is the policy that ABAC enforces which involves confidentiality and role.

In our model, as introduced in Defn. 5:, ψ represents the credential expression as in (Winsborough, Seamons, & Jones, 2000). In addition, we also create a set of release actions $\mathcal{RA}$ (Defn. 18) that the controller will perform depending on how ψ is satisfied. $\mathcal{RA}$ and ψ are created dynamically based on the request context the controller receives to ensure that its requirements for access match the resource being requested and the user's role. The $\mathcal{RA}$ may detail actions including but not limited to: logging, audit notifications, data redaction, data addition, or data modification. During trust negotiation, the controller attempts to minimize the risk present in the transaction by obtaining as many relevant credentials as possible until all of its requirements have been satisfied. The $\mathcal{RA}$ present in the model allows the controller to ascertain the amount of risk in the transaction and act accordingly to protect the resource, or otherwise add, modify, or remove access to portions of the

resource to reduce risk. This approach allows trust negotiation to obtain a higher success rate without compromising security.

4.3. Healthcare Example

In this section, we present a healthcare example that describes the interactions between the user (requestor) and the healthcare organization's controller involving the use of the trust profile as trust negotiation credentials, the validation of the trust profile's legitimacy, and the dissemination of new trust profile entries. A representation of Dr. Jane's current trust profile is shown in the inner box of Figure 4.4. Dr. Jane is a physician working at Family Health Center (FHC) and has previously obtained access to sensitive data at St. Francis Hospital (SFH). Her trust profile contains entries from both FHC and SFH. She possesses two X.509 identity certificates, one from each FHC and SFH displayed in the upper half of Dr. Jane's current trust profile in Figure 4.4. Each of the identity certificates has one or more attribute certificates attached to it, each attribute certificate describing one specific aspect of access to sensitive data under the physician role. Since she is an employee of FHC, her access to the EHR of FHC is unrestricted with improper data access being determined by local RBAC policies and security audits of the EHR's access logs. Since Jane is personally known to FHC and has had her personal identity manually vetted through FHCs hiring process, FHC endorses her trustworthiness by maintaining a current $AC_{Affiliation}$ certificate present in Jane's trust profile. Sensitive data from SFH was previously obtained through successful trust negotiation, resulting in additional trust profile entries. Jane's trust profile is stored remotely on trusted FHC servers and accessed by her security agent when needed for trust negotiation.

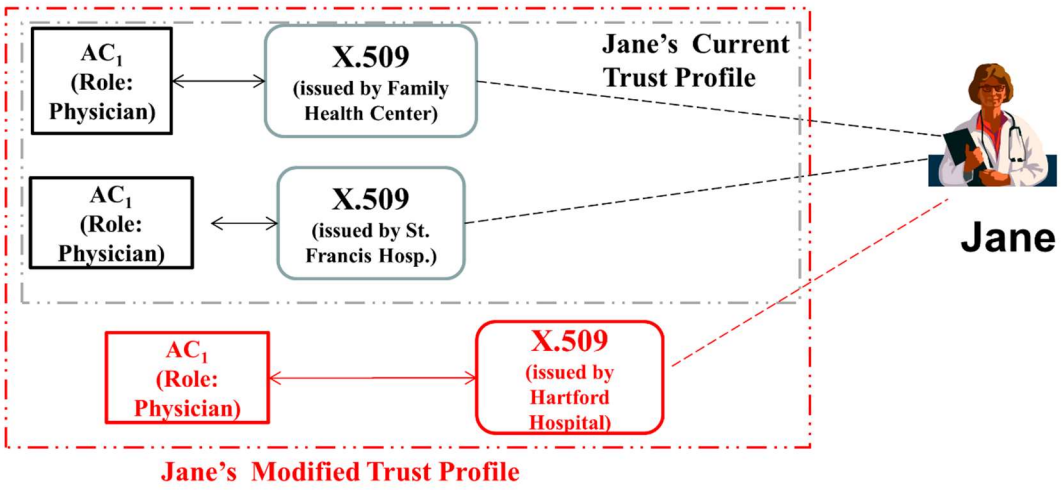

Figure 4.4. Dr. Jane's Trust Profile.

Jane is seeing her patient at FHC who recently had an EKG taken at the Henry Low Heart Center at Hartford Hospital (HH) and needs the patient's updated healthcare record with the EKG for follow up treatment. Jane has never had previous contact with HH for healthcare records, and HH has no previous contact with Jane or knowledge of her status as a physician with FHC. Jane is able to locate her patient's new EKG data through the use of a master patient index (MPI) and discovers that HH has a copy of her patient's EKG. Through the mHealth application on Jane's mobile device provided by FHC, Jane makes a request for trust negotiation to HH's controller to access her patient's EKG. The process of testing trust credentials starts in step 1 and continues in time from top to bottom in Figure 4.5.

Jane initiates a secure connection to HH's controller and creates an $\mathcal{RC}$. Recall from Defn. 16: that an $\mathcal{RC}$ consists of a *data resource DR*, the digital wallet tp_{DW} for the first round of trust negotiation, and the *Role r* that Jane would like to use to access the information. Jane builds the $\mathcal{RC}$:

- $DR = < DR^{EHR}_{Descriptor}, \ DR^{patient}_{ID} >$
- $tp_{DW} = < IC^{FHC}_{Issuer}, AC^{FHC}_{Affiliation} >$
- $\mathcal{RC} = < DR, tp_{DW}, r_{family\ physician} >$

Jane's mobile device communicates with her security agent $\mathcal{SA}_{Requestor}$ running on a server maintained by FHC to send the $\mathcal{RC}$, shown in step 1 in the upper left of Figure 4.5. $\mathcal{SA}_{Requestor}$ serves the purpose of offloading memory intensive and computationally expensive cryptographic operations required for trust negotiation from the mobile device as supported in surrogate trust negotiation (Sundelin, July 2003). Offloading management of trust profile disclosure to the security agent also improves security, as the trust profile will not be found on the mobile device in the case that it is lost or stolen. Recall that the security agent is responsible for managing the disclosure of Jane's credentials from her trust profile stored in the cloud. Jane selects a subset of her trust profile that describes her current employment with FHC under the family physician role. The mHealth application collects the specific certificates that describe the subset of the trust profile that Dr. Jane has selected and packages their certificate chains into a digital wallet tp_{DW}. The $\mathcal{RC}$ is then sent to the controller of HH's EHR, shown in step 2 of Figure 4.5.

HH's controller's security agent $\mathcal{SA}_{Controller}$ receives the $\mathcal{RC}$ and examines Dr. Jane's request to compile an appropriate set of credentials to determine whether Dr. Jane is trustworthy and should be granted access to the requested patient EKG. This collection of trust negotiation credentials that the controller requires to release the requested data is referred to as the credential expression, ψ. The credential expression ψ is generated, shown

in step 3 of Figure 4.5, based on $\mathcal{DR}_{type}$ and $r_{family\ physician}$. The controller decides that the request will be accepted if the requestor possesses the following credentials:

- The requestor has at least one credential $\mathcal{AC}_{Affiliation}$ indicating current employment and manual security vetting of the trust profile owner.
- The requestor has at least three credentials $\mathcal{AC}_{DataResourceConfidentiality}$ N indicating access to healthcare records of at least sensitivity level N.
- The requestor has at least three credentials $\mathcal{AC}_{DataResourceAccess}$ where *r* = *physician* indicating record access under the physician role.

Additionally, the request will be considered a lower risk transaction if the requestor possesses credentials indicating:

- The requestor has at least 3 credentials that indicate access to healthcare records of the patient listed in $\mathcal{DR}_{PD}$
- There is at least one credential indicating access to the patient's healthcare records within the last year $AC^{t}_{DataResourceAccess} = 2019$.

And the controller's release actions $\mathcal{RA}$:

- If the credential expression is satisfied but the additional requirements are not, the controller will release the data but will note the transaction in a high risk log and dispatch a notification to the controller's auditor that a high-risk transaction has occurred.
- If the credential expression is satisfied but only one of the additional requirements is met, the transaction is noted in a high risk log but no notification is sent to the auditor.

- If the credential expression is satisfied and both additional requirements are met, transaction is noted in low risk log.

The controller then begins the validation process for the certificates in the tp_{DW}. First, the controller requires proof that the sender of the certificates is also the owner of the certificates and that the entries in the trust profile describe the sender's access history. The controller determines ownership by sending a cryptographic challenge, shown in step 4 of Figure 4.5, encrypted with the public key listed in the associated identity certificates to the requestor, in this case Dr. Jane. Dr. Jane possesses the private key associated with the identity certificates, so her $\mathcal{SA}_{Requestor}$ is able to respond to the cryptographic challenge and signs the response with the private key before sending it back to the HH controller, shown in step 5. The HH controller, now satisfied that it is in communication with the trust profile's owner, begins the process of validating the information within the certificates (step 6). First, the controller examines the issuer signature of the leaf identity certificate. The controller retrieves the public key of the issuer from the next certificate in the chain in the CA certificate, in this case the signing certificate owned by FHC. The public key is used to retrieve the hash of the leaf identity certificate in the digital signature signed by FHC's CA certificate. The controller computes the hash of the leaf identity certificate and compares it to the CA's signed hash. If the hashes match, the controller knows that the data in the identity certificate is the same data signed by FHC if FHC's CA certificate is also valid and unaltered. Validation continues by validating FHC's CA certificate. The controller checks if FHC's CA certificate is in HH's local trusted certificate store. Since HH does not trust FHC directly, FHC's CA certificate is not in the certificate store so validation of the FHC CA certificate proceeds using the same process. The root medical

authority's certificate is retrieved, the signature of the FHC CA certificate is inspected, and the FHC CA certificate is found to be valid. The root medical authority's certificate is found in HH's local trusted certificate store, so the entire certificate chain is valid and Jane's identity certificates have been verified. The attribute certificates are validated with the same process, in addition to the controller checking to ensure each attribute certificate is associated with a valid X.509 identity certificate.

The digital wallet released to the controller thus far has only satisfied the affiliation requirement in the credential expression. The controller sends the SGP to Jane (step 7) listing remaining necessary credentials, including both those required for access and those required to consider the transaction low risk. In healthcare, the SGP is crucial to control access to sensitive medical data on patients by alerting the physician to the access history in the trust profile that is necessary to access the medical data. Without the SGP, the physician has no method to determine the type of health record access history the controller requires. It is necessary to alert the physician to the exact credentials required as the trust profile can become arbitrarily long and the physician must be able to complete the trust negotiation attempt quickly. Jane's trust agent receives the request (step 7). Jane reads the request (step 8) and decides to release the three latest records of access (step 9) to the patient's healthcare record in her trust profile and credentials indicating her confidentiality level. Her trust agent collects the certificates describing these access records, places them in a tp_{DW}, and sends them to the HH controller (step 10).

- $AC = < IC_{FHC}, ds_{FHC}, AC_{DataResourceAccess},\ 2020,\ u_{patient}, r_{physician}, dr_{EHR}, sys_{FHC} >$
- $AC = < IC_{FHC}, ds_{FHC},\ AC_{DataResourceConfidentiality},\ 2020, cl_R, dr_{patient}, sys_{FHC} >$

The controller receives Dr. Jane's credentials (step 11) and issues its cryptographic challenge to determine ownership (step 12). The trust agent answers the challenge (step 13). The controller checks that the credentials reference the correct patient, and checks the timestamps to determine if any of the accesses have occurred within the last year. Since the subset of Jane's credentials the controller holds now satisfy all of the controller's requirements (step 14), the controller decides to release the requested health data to Jane and considers the transaction to be low risk. The low risk status allows full data disclosure and the transaction is logged in a low-risk transaction log, fulfilling the ℛ@ requirements. The controller processes the EKG to be sent to Dr. Jane by converting it to a standard interchange format and preparing it for transfer using the FHIR standard. HH's controller requests that Dr. Jane either send an X.509 identity certificate signed by HH's controller or generate a new private/public key pair and send the new public key (step 15). Dr. Jane doesn't currently possess an X.509 identity certificate from HH so she generates the new key pair and sends the public key in a CSR as requested (step 16). HH's controller receives the public key and generates a new X.509 certificate, signs it, and creates attribute certificates describing the current access to the EKG (step 17). HH's controller then sends the EKG and the new trust profile entries to Dr. Jane over the secure connection (step 18). Dr. Jane receives the trust profile entries, adds them to her trust profile, and is now able to examine the EKG (step 19). The trust negotiation process is now complete and the connection is terminated. Should Dr. Jane already possess an X.509 identity certificate from HH, the process remains the same except HH only generates the attribute certificates and associates them with the already created identity certificate. Dr. Jane's trust profile

now encompasses the entire trust profile present in Figure 4.4, including the certificates issued by HH.

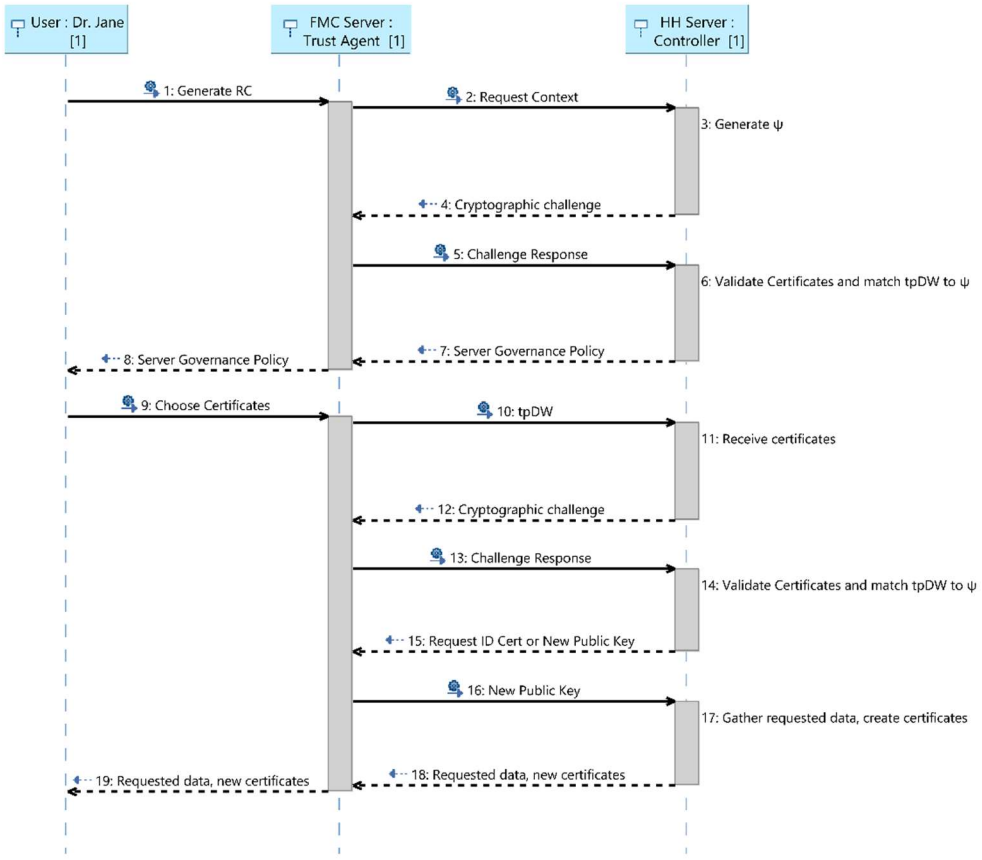

Figure 4.5. Healthcare Example Sequence Diagram.

To illustrate the flexibility of the controller's requirements, suppose that Jane is unable to produce a credential indicating previous access of the patient's health records because her patient typically sees another physician at the same hospital. In this case, the controller's requirements for the data are met, but the transaction is considered by the controller to be a higher risk because the physician lacks a history indicating treatment of the patient in the past. The healthcare data is still released to Jane since she has indicated a

previous history of healthcare data access under the physician role and current affiliation with a healthcare organization, but since the controller is unable to obtain assurance that Jane has had past interactions with the patient, the transaction is logged in a high risk audit log and the controller's security auditor receives a notification that the high risk transaction has occurred. This allows Jane timely access to the data with a reasonable level of assurance that the transaction is appropriate while allowing the auditor to react swiftly to the transaction if a mistake has occurred.

4.4. Related Work

The security and flexibility of trust negotiation has resulted in a variety of adaptations to the healthcare domain and online business, in conjunction with mobile devices. In (Ryutov, Zhou, Neuman, Leithead, & Seamons, 2005), the authors create a framework for adaptive trust negotiation targeting trust establishment between unknown users and online businesses through integrating TrustBuilder and the GAA-API. Trust is established utilizing credentials accumulated through past purchases online, with users being authorized to make larger purchases online if they are able to present a history of past successful purchases. Users are assigned a score by the business' controller based on both the quantity and dollar amount of past purchases and the score determines whether the current transaction can be trusted to be fulfilled by the user. The required level of trust also fluctuates depending on the number of requests received by a single requestor to guard against slowing the system with a denial-of-service attack composed of repeated failed trust negotiation attempts. This approach of tracking users' past purchases is similar to our approach of accumulating a set of historical access to sensitive healthcare data, which provides an accumulating set of credentials. Our approach contrasts with this effort by

allowing for more depth in which credentials are specified, which properties of the credentials are most relevant, customizing the credential requirements based on the user's role, and customizing the controller's response to a trust negotiation request.

In the healthcare domain, (Mavridis, Georgiadis, Pangalos, & Khair, 2001) provides a methodology for applying access control to EHR data through the use of attribute based access control utilizing a certificate system similar to the one presented here. Unlike our approach where the user possesses the certificates detailing attributes directly, the authors use an identity based approach. Each user obtains an identity certificate and is also provided with an appropriate set of short-lived attribute certificates. These certificates provide credentials that the access control policy reads to determine sensitive data access, including the concept of a user location defined by: the actual device the user operates to initiate the request, the user's administrative domain that determines how it reacts to the request based on group policies (e.g., different departments within the hospital), and a context parameter that detects a need-to-know requirement (e.g., physicians may only read data on patients they have been assigned). Access is based on a tuple consisting of the user's role and location, which grants permissions to access a data set under an access mode. This approach mirrors our approach with regards to the usage of credentials encoded in identity and attribute certificates but lacks methods of accumulating credentials over a period of time, which is a key feature of our approach and lacks the adaptiveness of providing multiple levels of trust which we also offer.

The application of trust negotiation for healthcare was introduced in (Vawdrey, Sundelin, Seamons, & Knutson, 2003). In this paper, the authors describe an EHR capable of receiving requests for trust negotiation and generating a policy that releases the

underlying medical data. The controller responds to requests for health data only by requesting the physician's medical license which is provided by a medical association. While our approach is also healthcare focused and depends on the user possessing a verifiable set of credentials, we have expanded the set of credentials beyond the usage of a medical license to a complete set of access history in the trust profile. Additionally, the controller is capable of adapting the security policies to both the type of request and the requestor.

Surrogate trust negotiation (Sundelin, July 2003) is also provided as a method for adapting trust negotiation to the healthcare field since healthcare data is often accessed via mobile device. The more cryptographically intensive functions such as validating certificates and sending responses to cryptographic challenges are offloaded to a trusted server provided by the hospital that employs the physician. We have incorporated the concept of a trust agent introduced in this paper into our model, allowing a trust profile owner to determine the way that the trust profile is stored. The communication between the user's trust agent, which manages trust profile credential disclosure for the user, and the controller's trust agent, which manages the controller's own credential disclosures and forwards an SGP to the user to communicate which trust profile disclosures are required, forms the basis for credential disclosure in our approach.

The application for healthcare is further enhanced in (Elkhodr, Shahrestani, & Cheung, 2011) by creating a trust negotiation protocol for verifying access to remote healthcare sensors placed in the home of the patient. The Ubiquitous Health Trust Protocol (UHTP) is created to allow physicians to authorize themselves, the device they use, and the remote sensors they access to retrieve up to date healthcare data, allowing the patient

to be monitored continuously from a healthcare professional from the comfort of their own home. Similar to our approach, the UHTP defines multiple credentials, building trust if the user is: authenticated with the local EHR, utilizing an approved mobile device, and located within a certain distance of the remote sensors being accessed. In comparison, our approach is different in three ways: it is more adaptive and allows multiple levels of trust to be attained depending on the credentials presented; contains a more expansive set of credentials consisting of the user's entire sensitive data access history; and, doesn't require traditional username/password authentication that necessitates previous credential registration.

Chapter 5
Dynamic Adaptive Trust Negotiation Framework

In this chapter, we describe the dynamic adaptive trust negotiation framework that includes all of the infrastructure that is necessary to design and implement the trust profile concepts from Chapter 3 and the model of Chapter 4 into a framework to realize trust negotiation. The discussion of the framework in this chapter supports Contribution D: Dynamic Adaptive Trust Negotiation Framework by:

1. defining and describing all of the necessary concepts that are required to implement the model of Chapter 4 using the resource in Defns. 6a and 6b;
2. augmenting the discussion of every concept to customize the discussion of each concept which essentially constitutes a way that the resource concepts can be transitioned into FHIR resources; and,
3. illustrating each of the concepts by continuing and extending the example presented in Section 4.3 of Chapter 4.

The main concepts of the dynamic adaptive trust negotiation framework are:

- *Security objects* (*sec objects*) that are created based on the data requested, the trust profile entries required, and the validated trust profile entires exchanged during trust negotiation that fulfill the trust profile requirements.
- *Security Metadata* that is divided into four concepts: *system security metadata, resource type security metadata, resource security metadata,* and *consent metadata..*

- *Security Object Structure* that creates a tree for each *Sec object* that contains the specific credentials that must be presented to the controller.
- *Request Resolution* which describes how multiple *Sec objects* are combined to ensure security at each identified security level from the four *Security Metadata* concepts.
- *Controller Configuration* which describes the way that the controller is configured to accept certain collections of credentials on a per-role basis.

This chapter consists of 2 sections. Section 5.1 discusses the contents, structure, and useage of a security object in detail. Section 5.2 discusses the configuration of a security object and the metadata, specified in JavaScript Object Notation (JSON) format, required to build a security object. All of the concepts discussed in both Sections 5.1 and 5.2. support expected Contribution C: Dynamically Generated Adaptive Access Control Policies and Contribution D: Trust Negotiation Development Framework.

5.1. Security Objects

In this section, we introduce the concept of a security object*, Sec object*, which acts as a repository to capture the relationship between the data requested, the trust profile entries required, and the validated trust profile entries exchanged during trust negotiation that fulfill the trust profile requirements. Recall from Chapter 4 that the trust profile's attribute certificate types track: the associated identity certificate, the attribute certificate issuer, and a timestamp. The attribute certificate types are: data resource access certificates ($\mathcal{AC}_{DataResourceAccess}$), affiliation certificates ($\mathcal{AC}_{Affiliation}$), data resource confidentiality certificates ($\mathcal{AC}_{DataResourceConfidentiality}$), and system confidentiality certificates ($\mathcal{AC}_{SystemConfidentiality}$); see Defns. 14a, 14b,

14c, and 14d, respectively. The data resource $\mathcal{DR}$ of a particular system $\mathcal{S}$ requires a set of access certificates $\mathcal{AC}$ that provide metadata on the role the user possessed during the access, a $\mathcal{DR}_{\mathcal{ID}}$, and the $\mathcal{S}_{\mathcal{ID}}$ representing system $\mathcal{S}$ (e.g., an EHR or a FHIR server in the healthcare domain) that serviced the originating request. Affiliation certificates denote that a user is a current employee with a trusted healthcare provider which implies a thorough manual background check as part of the pre-employment process. The data resource confidentiality certificate provides the confidentiality level of the resource accessed $\mathcal{DR}_{\mathcal{ID}}$, within the system $\mathcal{S}_{\mathcal{ID}}$. The system confidentiality certificate describes the highest level of confidentiality that the certificate subject has accessed on the system $\mathcal{S}_{\mathcal{N}ame}$ with $\mathcal{S}_{\mathcal{ID}}$.

The controller (Defn. 3) first receives a request context $\mathcal{RC}$ (Defn. 16) consisting of the requested $\mathcal{DR}$ (Defn. 6b), the digital wallet (Defn. 15), and role r (Defn. 8) for trust profile based trust negotiation. The controller creates one or more *Sec objects*. Each *Sec object* is responsible for matching verified trust profile credentials to security requirements at a pre-defined security level that corresponds to collections of sensitive data. These security levels are identified when trust profile support is added to a server's authentication and authorization options. The first security level encompasses the entire server, with each lower security level configured with stricter requirements for a more specific subset of data. In order to obtain access to the requested object, the requestor's trust profile entries must satisfy the requirements of the *Sec objects* at each level. We define a single *Sec object* as being satisfied when a subset of the trust profile credentials shared with the controller by the requestor $\mathcal{R}$ contains the Access History Properties $\mathcal{AHP}$ (Defn. 9) with the values required by the Controller Configuration to be introduced in Section 5.2. Security metadata

is divided into one of four identified levels, whose various access rules are combined utilizing the process described in Section 5.1.5 to form one credential expression for the entire trust negotiation process. Recall from Chapter 2 Section 3 that FHIR categorizes resources into one of many types of resources. These four security levels are:

- *System security metadata* to be discussed in Section 5.1.1 refers to the requirements that the controller must observe in the requestor's trust profile to gain access to any resources on $\mathbb{S}$.
- *Resource type security metadata* to be discussed in Section 5.1.2 refers to the protection of an individual type of resource.
- *Resource security metadata* to be discussed in Section 5.1.3 refers to the protection of an individual resource instance on $\mathbb{S}$ and the data within the resource.
- *Consent metadata* to be discussed in Section 5.1.4 refers to the ability of a patient to describe which healthcare providers may access each resource that describes the patient.

In support of trust profile integration, we have created a front end to a RESTful implementation to enable the creation of a credential expression, through the creation of *Sec objects*, whose access rules are created based on a pre-defined configuration, to be specified in Section 5.2, and reactive to the requestor's role and the trust profile entries received. This metadata describes the properties needed in the requestor's trust profile to obtain authorization to a resource on a per-role basis. The modifications support the creation of an adaptive credential expression that accepts multilple sets of trust profile credentials that result in multiple levels of trust. Depending on the level of trust established,

the configuration specifies release actions $\mathcal{RA}$ for each set of credentials that may satisfy the generated credential expression.

Each *Sec object* is organized into a tree structure as illustrated with the $\mathcal{Sec}_{\mathcal{Resource\ Type}}$ example in Figure 5.1 detailing the specific credentials that must be presented to the controller on a per role basis, as well as release actions required for the release of the resource based on which parts of the credential expression are satisfied. The root of the tree contains an identifier that defines the type of *Sec object*, depicted in Figure 5.1 as the subscript in the purple rectangle on the left side. The $\mathcal{Sec}_{\mathcal{System}}$, $\mathcal{Sec}_{\mathcal{Resource\ Type}}$, $\mathcal{Sec}_{\mathcal{Resource}}$, and $\mathcal{Sec}_{\mathcal{Consent}}$ objects protect, respectively, the involved system $\mathcal{S}_{\mathcal{ID}}$, an identified data resource type $\mathcal{DR}_{\mathcal{Type}}$ of that system, an associated data resource $\mathcal{DR}_{\mathcal{ID}}$ (e.g., an ID of the patient FHIR resource), or the consent of the data's owner (e.g., a patient). A *Sec object* ID, represented as the gray oval in the bottom left of Figure 5.1, notes a unique identifier for the actual instance of the resource being protected (e.g., a system ID, the resource type's name, a numeric identifier for the individual resource, etc.). The next level of the tree represented by the branches connected to the $\mathcal{Sec}_{\mathcal{Resource\ Type}}$ rectangle in Figure 5.1, provides a supported list of roles capable of retrieving data of the requested *Sec object*. For example, a $\mathcal{Sec}_{\mathcal{System}}$ object contains a complete listing of all of the roles that are able to access any sensitive data protected by the controller, whereas $\mathcal{Sec}_{\mathcal{Resource\ Type}}$ for an Observation FHIR resource will only contain the roles that are capable of accessing a resource of the Observation type. Each role contains subtrees representing a set of $\mathcal{AHP}$s, shown as green rectangles connected to the roles in Figure 5.1, that the controller must request from the requestor's trust profile to grant access if the requestor is assuming that role. Each role is configured

with multiple sets of $\mathcal{AKP}$s with differing requirements and release actions. The *Sec object*'s security requirements are satisfied if one set of $\mathcal{AKP}$s specified in the configuration has been matched to the presented trust profile credentials (e.g., the *Sec object* requires trust profile credentials noting access to the patient's record within the last year, and a trust profile credential indicating access to the patient's record within the last year is presented). The existence of multiple sets of $\mathcal{AKP}$s allows more flexibility in the ability to build trust between the requestor and controller by requesting multiple combinations of $\mathcal{AKP}$s, which provides a baseline for $\mathcal{AKP}$s that must be present for the controller to trust the requestor with the release of the requested resource object.

Each set of $\mathcal{AKP}$s has an optional set of release actions ($\mathcal{RA}$s) represented by the blue rectangles attached to the $\mathcal{AKP}$s that describes ancillary actions the controller must take to approve the satisfaction of the $\mathcal{AKP}$ requirement by a credential in the requestor's trust profile. The $\mathcal{RA}$ for an $\mathcal{AKP}$ optionally has: potential additions, modifications, or redactions of the resource before release to the requestor; or specifies side effect actions such as noting the release of the resource at certain risk levels in a multi-level audit log and dispatching audit notifications to the healthcare organization's local security auditor for immediate review. Additions to the resource include contextual data not requested but necessary to understand the resource, e.g., a program for reading X-Ray scans. Modifications to the resource include changes such as translating embedded data into a standard format. Redactions may occur if the requestor's credentials meet a trust level sufficient to access parts of a resource, but not the entire resource. In this case, the sensitive data is redacted, allowing the requestor to obtain the subset of useful data that the requestor

is authorized to access. Integrating an $\mathcal{RA}$ into a resource is a method that the controller uses to increase the rate of trust negotiation success and disseminate requested PHI without compromising patient security. The remainder of this section consists of Sections 5.1.1 to 5.1.5, which correspond to the concepts: System Security metadata, Resource Type Security metadata, Resource Security metadata, Consent metadata, and Request Resolution.

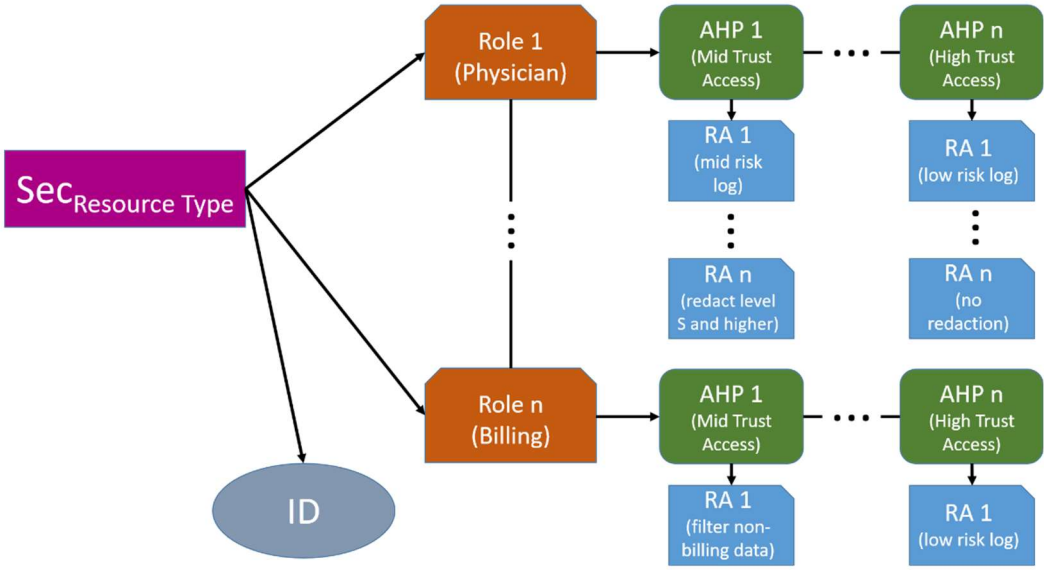

Figure 5.1. $\mathcal{Sec}_{\mathcal{Resource\ Type}}$ Object Example Structure.

5.1.1. System Security Metadata

System security ($\mathcal{Sec}_{\mathcal{System}}$) *metadata* refers to the requirements the controller must observe in the requestor's trust profile to gain access to any resources protected by the system. This may include a valid affiliation certificate (Defn. 14b) denoting current employment at a trusted healthcare provider and at least one data resource access certificate (Defn. 14a) describing access to a resource under the role requested for the current trust negotiation. *System security* also encompasses the overall highest security clearance the

user has been granted on a specific system. As specified in the trust profile model, an identity certificate in a trust profile may have a system confidentiality attribute certificate (Defn. 14d) attached to it that records the highest security clearance previously granted to the trust profile owner by the specific system that signed it. This certificate is replaced with a newer certificate listing a higher clearance in the event that the controller grants the requestor access to a resource with a higher listed security clearance than the clearance listed in the requestor's system confidentiality attribute certificate. The requestor may be assigned higher security clearances by the controller depending on which trust profile entries the requestor sends to satisfy the generated credential expression. An $\mathcal{AC}_{SystemConfidentiality}$ certificate previously digitally signed by the controller may be provided during negotiation to claim a previous confidentiality level assigned by the controller. A system confidentiality level may be assigned based on the perceived damage caused by a potential unauthorized leak of the requested data. A requestor that meets the confidentiality requirements for portions of the requested resource, but not the entire resource, may result in a *release action* ($\mathcal{RA}$) (Defn. 18) that causes the controller to filter data of higher sensitivity from the resource before sending it to the requestor.

The system security metadata of the framework can be integrated through a front end into FHIR resources and a corresponding HAPI FHIR server implementation in a number of steps. $\mathcal{Sec}_{System}$ represents the base requirements for any request made to the HAPI FHIR server. When trust negotiation with a HAPI FHIR instance is requested, the $\mathcal{Sec}_{System}$ object is created based on the role the user chooses to represent the set of trust profile requirements to access the HAPI FHIR API. Any request to the HAPI FHIR instance that does not satisfy the $\mathcal{Sec}_{System}$ object is unable to retrieve any healthcare data from the

underlying EHR. Figure 5.2 shows an example configuration of a $\mathcal{S}ec_{\mathcal{S}ystem}$ object for the Primary Care Physician, Nurse, and Accident and Emergency doctor roles. These configurations require that each role has a current affiliation certificate (Defn. 14b). Additionally, with no other credentials or overrides from the other *Sec objects*, the request would be logged as a high risk transaction and an audit notification would be dispatched to the auditor account located at "security@bmi9.engr.uconn.edu".

```
{
    "type": "system",
    "id": "bmi9.engr.uconn.edu/Hackathon/UCONN-FHIR/",
    "446050000": [ // PCP - Primary care physician
        {
            "affiliation": {
                "period": {
                    "to": "current"
                },
                "release-actions": [
                    {
                        "log": "high",
                        "audit": "security@bmi9.engr.uconn.edu"
                    }
                ]
            }
        }
    ],
    "106292003":  [ // Nurse
        {
            "affiliation": {
                "period": {
                    "to": "current"
                },
                "release-actions": [
                    {
                        "log": "high",
                        "audit": "security@bmi9.engr.uconn.edu"
                    }
                ]
            }
        }
    ],
    "309294001": [ // Accident and Emergency doctor
        {
            "affiliation": {
                "period": {
                    "to": "current"
                },
                "release-actions": [
                    {
                        "log": "high",
                        "audit": "security@bmi9.engr.uconn.edu"
                    }
                ]
            }
        }
    ]
}
```

Figure 5.2. An Example $\mathcal{S}ec_{\mathcal{S}ystem}$ Configuration.

5.1.2. Resource Type Security Metadata

Resource type security ($\mathcal{S}ec_{\mathcal{R}esource\ \mathcal{T}ype}$) metadata refers to the protection of all of the resources of a given type (e.g. resource types in the FHIR standard, which are further divided into foundation, base, clinical, financial, and specialized resource types (HL7 International, 2019)). Each resource type possesses an associated security object that describes the credentials that must be presented by the requestor for the controller to release a resource of the given type. The required credentials are described by a series of *Access History Properties ($\mathfrak{AHP}$)*, each describes a single property of access to a sensitive resource. These credentials are also organized within the security object by the role the requestor assumes for the given trust negotiation transaction. Recall that (Sanzi et al., 2017) specifies that in the initial request for trust profile based trust negotiation, the requestor specifies the role (e.g., family physician, emergency room physician, nurse, billing agent, front desk secretary, etc.) to be assumed for the purposes of negotiation. The controller filters the resources a requestor of a given role accesses based on the perceived needs of a role. For example, a physician role will be allowed to access clinical resources (e.g., summary, diagnostics, medications, care provisions, request and responses) whereas a front desk secretary may be limited to the patient resource (describing the patient's demographic data) under the base category. The role specified by the requestor will also affect the proof the controller requests for assurance of the requestor's membership of the specified role. A family physician requesting a patient's clinical resources may be asked to provide credentials from the trust profile indicating a historical access to the patient's clinical resources whereas a front desk secretary may only have to provide proof of employment (affiliation) via a desk secretary role to access a patient's demographic data.

The resource type security metadata of the framework can be integrated through a front end into FHIR resources and a corresponding HAPI FHIR server implementation in a number of steps. $\mathcal{Sec}_{\mathcal{Resource\ Type}}$ represents the requirements to access an object of the requested type from the HAPI FHIR instance. If the requestor were to request a FHIR *Observation* resource from the controller, the controller retrieves a $\mathcal{Sec}_{\mathcal{Resource\ Type}}$ for the *Observation* resource that contains the set of Access History Properties required in the trust profile to obtain access to any resource of an *Observation* type. If a requestor wishes to retrieve an *Observation* resource, at this level the requestor must: satisfy the $\mathcal{Sec}_{\mathcal{System}}$ for HAPI FHIR API access and satisfy the $\mathcal{Sec}_{\mathcal{Resource\ Type}}$ for *Observation* resource access. Figure 5.3 shows an example $\mathcal{Sec}_{\mathcal{Resource\ Type}}$ configuration for a $\mathcal{Sec}_{\mathcal{Resource\ Type}}$ object for an *Observation* resource. The displayed role, for the physician, is required to show a current affiliation (Defn. 14b) as well as satisfy one of the two sets of properties noted under the "properties" array. Note that more roles are listed at this level in the JSON structure in Figure 5.3 in the same manner as in Figure 5.2 but are omitted for brevity. The physician's properties array contains two $\mathfrak{QKP}$ (Defn. 9) sets: one at the top whose satisfaction results in a high amount of trust, and one at the bottom whose satisfaction results in a mid level of trust. The high trust $\mathfrak{QKP}$ set requires 3 (quantity property) trust profile entries within 3 years of the current date (period property), with the role of physician (role property), with the patient's ID for those trust profile entries matching the one for the Observation (patient property), and the trust profile entries must be Observations (resource-type property) with a clearance of TS (resource-clearance property). Should all of these requirements be met by the presented trust profile, the requestor gains a high level of trust, the transaction does not require an audit, and the transaction is logged as a low risk (release-actions property). Conversely, the mid-level

trust $\mathcal{QKP}$ set only requires one entry within the last year, with the patient's ID, only requires access to the Patient resource (containing patient demographics), requires only a resource-clearance of S, and results in logging the transaction as a mid level risk.

```
{
    // the type of this Sec object is SecResource Type
    "type": "type",
    "id": "Observation",

    "446050000": { // Physician
        "affiliation": {
            "period": {
                "to": {
                    "type": "current"
                }
            }
        },
        "properties": [
            { // Establishes high trust
                "quantity": 3,
                "period": {
                    "from": {
                        "type": "relative",
                        "value": 3,
                        "unit": "year"
                    },
                    "to": null
                },
                "role": "446050000", // PCP - Primary care physician
                "patient": "request_id",
                "resource-type": "Observation",
                "resource-clearance": "TS",
                "release-actions": {
                    "audit": null, // cancel any audit notification requirements, this is a high trust transaction
                    "log": "low" // log in a low risk level log
                }
            },
            { // Establishes mid level trust
                "quantity": 1,
                "period": {
                    "from": {
                        "type": "relative",
                        "value": 1,
                        "unit": "year"
                    },
                    "to": null
                },
                "role": "446050000", // PCP - Primary care physician
                "patient": "request_id",
                "resource-type": "Patient",
                "resource-clearance": "S",
                "release-actions": {
                    "audit": null, // cancel any audit notification requirements, this is a mid trust transaction
                    "log": "mid" // log in a mid risk level log
                }
            }
        ]
    },
```

Figure 5.3. An Example $\mathcal{SecResource\ Type}$ Configuration.

5.1.3. Resource Security Metadata

Resource security ($\mathcal{S}ec_{\mathcal{R}esource}$) metadata protects an individual resource object on the a server and the data within the object. The resource $\mathcal{S}ec_{\mathcal{R}esource}$ provides security data for is identified within the $\mathcal{S}ec_{\mathcal{R}esource}$ by a matching the identifier presented in the Data Resource (Defn. 6b) of the Request Context (Defn. 16). $\mathcal{S}ec_{\mathcal{R}esource}$ is similar to $\mathcal{S}ec_{\mathcal{R}esource\ \mathcal{T}ype}$ with the exception that it protects an individual resource instance as contrasted to an entire collection of resources of a certain type, increasing the granularity with which a resource is protected. When the request for a resource through trust negotiation is first received by the controller, the $\mathcal{S}ec_{\mathcal{R}esource}$ object attached to the requested object is retrieved by matching the object's ID to the $\mathcal{S}ec_{\mathcal{R}esource}$ ID.

The resource security metadata of the framework can be integrated through a front end into FHIR resources and a corresponding HAPI FHIR server implementation. $\mathcal{S}ec_{\mathcal{R}esource}$ represents the requirements to access the specific FHIR resource object identified in the initial data request. The $\mathcal{S}ec_{\mathcal{R}esource}$ is retrieved via a FHIR ID and provides security for an individual FHIR resource instance. When a requestor requests a resource, at this level the requestor must satisfy: the $\mathcal{S}ec_{\mathcal{S}ystem}$ for HAPI FHIR API access, the $\mathcal{S}ec_{\mathcal{R}esource\ \mathcal{T}ype}$ for *Observation* resource access, and the $\mathcal{S}ec_{\mathcal{R}esource}$ object for access to the specific resource requested. Figure 5.4 shows an example configuration for a specific Observation resource, identified by the EHR using the FHIR URL to access the Observation with an ID of 25.

```
{
    "type": "resource",
    "id": "bmi9.engr.uconn.edu/Hackathon/UCONN-FHIR/Observation/25",
    "446050000": { // Physician
        "affiliation": {
            "period": {
                "to": {
                    "type": "current"
                }
            }
        },
        "properties": [
            { // Establishes high trust
                "quantity": 3,
                "period": {
                    "from": {
                        "type": "relative",
                        "value": 3,
                        "unit": "year"
                    },
                    "to": null
                },
                "role": "446050000", // PCP - Primary care physician
                "patient": "request_id",
                "resource-type": "Observation",
                "resource-clearance": "TS",
                "release-actions": {
                    "audit": null, // cancel any audit notification requirements, this is a high trust transaction
                    "log": "low" // log in a low risk level log
                }
            },
```


Figure 5.4. An Example $\mathcal{S}ec_{\mathcal{R}esource}$ Configuration.

5.1.4. Consent Security Metadata

Consent security ($\mathcal{S}ec_{\mathcal{C}onsent}$) metadata allows for a data owner to provide input as to which requestors may access each resource owned. A $\mathcal{S}ec_{\mathcal{C}onsent}$ is retrieved during trust negotiation by matching the ID of the owner described by the resource with the ID of the $\mathcal{S}ec_{\mathcal{C}onsent}$ object. Additionally, a $\mathcal{S}ec_{\mathcal{C}onsent}$ object provides support for listing an arbitrary list of trust profile identity certificates whose owners can access the object regardless of the requirements listed in the *Consent* object. This allows explicit access to the object should the data owner know of a user that should be able to access the object, even if the organization maintaining the controller does not. Identity certificates are uniquely identified by a combination of issuer and serial number, and a public key must be listed for the requestor to authenticate to.

The consent security metadata of the framework can be integrated through a front end into FHIR resources and a corresponding HAPI FHIR server implementation. A $\mathcal{S}ec_{\mathcal{C}onsent}$ of type Patient allows the patient whose medical data is described by the requested resource to have input as to which healthcare providers may access the resource. Our $\mathcal{S}ec_{\mathcal{C}onsent}$ security object is based on the principles of patient consent (The Office of the National Coordinator for Health Information Technology, 2019) outlined by The Office of the National Coordinator for Health Information Technology (ONC). Patient consent methods allow patients to consent to HIE among multiple healthcare providers by allowing patients to note when and how their health data is shared whether their health data is shared for treatment, bill payment, or general healthcare operations. Our patient consent object can override other security objects when present, allowing the patient to have final authority over the disclosure of the health record. The $\mathcal{S}ec_{\mathcal{C}onsent}$ object is built by the patient and attached to the patient's records within a FHIR system, allowing the patient to provide input as to which trust profile credentials are necessary during trust negotiation for the release of the patient's FHIR resources.

The patient interacts with the $\mathcal{S}ec_{\mathcal{C}onsent}$ object via a patient portal provided by the healthcare organization maintaining the FHIR server. The patient portal follows the ONC's meaningful consent guidelines (The Office of the National Coordinator for Health Information Technology, 2018) and describes the patient's choices as well as the implication of their options regarding what data will be released to which types of providers under different circumstances. The patient portal interface provided by the healthcare organization presents multiple options that cover different use cases along with descriptions for which healthcare providers have access to the patient's FHIR resources

depending on the options chosen. This simplifies the selection process for a patient, allowing the patient to fully comprehend the implications of each choice without requiring a deep understanding of trust profiles or trust negotiation. At the healthcare provider's discretion, more granular interfaces can be made available to the patient should the patient have the knowledge to construct more detailed $Sec_{Consent}$ objects. The $Sec_{Consent}$ object contains the same format as the $Sec_{Resource}$ and $Sec_{Resource\ Type}$ objects with the restriction that a $Sec_{Consent}$ object is only attached to a FHIR resource via ID if the resource's patient identifier matches the identifier of the patient creating the $Sec_{Consent}$ object. Additionally, the patient may include multiple instances of a healthcare professional's public key from a trust profile identity certificate. This allows the patient to identify a healthcare professional as being able to access the patient's healthcare records if the patient has a pre-existing relationship. If an identity certificate is listed as a potential credential to gain access to a FHIR resource, the healthcare professional attempting to access the resource proves ownership of the public key by proving ownership of the associated private key. This is done by digitally signing a message with the private key during trust negotiation in accordance with public key infrastructure. This feature allows patients to name specific healthcare workers that should be able to access their records, such as in the case of being treated by multiple physicians known to the patient, each providing one specific aspect of treatment.

Figure 5.5 shows an example $Sec_{Consent}$ configuration that belongs to a patient with an ID of 13. Although the patient has not specified any CHP sets, the patient has explicitly allowed a physician with the matching certificate listed in the "direct-consent" object to access any healthcare record that describes the patient. The physician can satisfy the $Sec_{Consent}$ object with this configuration by presenting the certificate with the specified issuer and

serial number and responding to a cryptographic challenge constructed with the listed public key. A successful response indicates that the requestor owns the listed identity certificate. This allows patients to formally specify their own physicians or other healthcare professionals as having access to their healthcare records. Although the properties list is empty in this example, patients can specify constraints on their healthcare data by requiring trust profile entries utilizing the same format for the properties objects as in the examples listed in Figures 5.2 through 5.4.

```
{
    "type": "consent",
    "id": "13",
    "446050000": { // Physician
        "affiliation": {
            "period": {
                "to": {
                    "type": "current"
                }
            }
        },
        "properties": [],
        "direct-consent": [
            {
                "issuer": "C=US,ST=CT,O=Gino's Hospital,CN=trustnegotiation.ginoshospital.com,
                           emailAddress=trustadmin@trustnegotiation.ginoshospital.com",
                "serial": "103929",
                "public-key": {
                    "algorithm": "rsaEncryption",
                    "modulus": "00:dc:d0:56:89:d0:47:f3:53:71:bb:01:f0:ca:4c:
                                e2:32:2b:48:5b:29:b4:05:dd:4c:84:a5:b0:84:09:
                                a0:bc:3a:6e:41:69:5d:89:12:91:72:c8:24:1a:39:
                                69:8a:b1:c7:e3:83:c7:bc:32:5c:76:4d:dc:be:68:
                                fe:71:25:ff:49:37:b4:23:f0:16:e1:be:b8:84:b0:
                                52:52:27:fd:c5:01:16:47:ce:11:0d:0e:a3:68:f5:
                                fb:4c:0e:0b:15:98:81:d1:89:80:ef:3c:8d:4a:14:
                                d5:ed:56:fe:33:d7:45:6e:62:d0:3d:69:02:9b:dc:
                                0d:a2:3a:ff:0f:e5:01:7c:49:ea:69:33:01:b2:32:
                                ab:23:d6:72:ba:b9:ba:94:2e:e7:bc:08:25:14:4f:
                                52:44:76:d9:cb:06:10:c7:a9:cd:f8:88:0a:58:d9:
                                97:c3:22:07:c6:68:e8:51:ff:c5:00:fe:d4:c8:cd:
                                c3:f9:c0:54:20:9c:43:01:26:20:26:df:bc:cd:66:
                                30:74:96:a4:66:42:9d:6e:1d:35:8e:9d:3d:2d:df:
                                fb:f4:8e:25:5e:d1:a0:91:6e:a0:05:21:3a:57:e9:
                                dc:4f:67:85:77:77:7a:5d:8f:45:9a:5e:fb:a2:29:
                                1f:ef:d6:4d:40:e1:06:7d:eb:ef:65:54:dd:0f:d6:
                                95:a9",
                    "exponent": "65537"
                }
            }
        ]
    }
}
```

Figure 5.5. An Example $\mathcal{S}_{ec_{Consent}}$ Configuration.

5.1.5. Request Resolution

Conceptually, each $\mathcal{QHP}$ listed in a *Sec object* represents an entry required to exist in the requestor's trust profile that proves successful, secure handling of the type of *Sec object* by the role. The healthcare organization that shares the PHI is responsible for determining the $\mathcal{QHP}$s necessary to determine whether a requestor is trustworthy. A requestor making a request under a family physician role for their patient's EHR data located at a remote healthcare organization could result in the following requested $\mathcal{QHP}$s and $\mathcal{RA}$s:

- $\mathcal{Sec}_{System}$: Affiliation with any healthcare provider ($\mathcal{RA}$: log as high risk).
- $\mathcal{Sec}_{Resource\ Type}$: Past access to a resource of the same type within the last year ($\mathcal{RA}$: reduce log level to medium risk, redact resource data with sensitivity: S or higher).
- $\mathcal{Sec}_{Resource}$: Optional: Past access to a resource belonging to the patient within the last two years ($\mathcal{RA}$: reduce log level to low risk, audit notification not required, no redaction required).
- $\mathcal{Sec}_{Consent}$: Affiliation with a listed healthcare provider ($\mathcal{RA}$: notify patient of access through a healthcare portal).

The request resolution of the framework can be integrated through a front end into FHIR resources and a corresponding HAPI FHIR server implementation. The *Sec object* structure identifiers and $\mathcal{QHP}$ sets represent resources in the FHIR standard. The $\mathcal{Sec}_{System}$ identifier is the FHIR RESTful URL where trust negotiation is initiated with the controller. The $\mathcal{Sec}_{Resource\ Type}$ identifier is a FHIR resource category such as an *Observation*, *Patient*, or

MedicationStatement. The $\mathcal{Sec}_{\mathcal{R}_{esource}}$ identifier is a FHIR ID that uniquely identifies the FHIR resource. The $\mathcal{Sec}_{\mathcal{C}_{onsent}}$ identifier is the ID of the patient whose healthcare data is represented by the FHIR object. $\mathfrak{QHP}$ (Defn. 9) sets describe trust profile entries noting access to sensitive healthcare data, such as the type of data accessed, when the data was accessed, and the specific patient whose data was accessed.

When a request for trust negotiation is initially received, the controller first retrieves the metadata utilizing the Request Context (Defn. 16) for each of the four *Sec objects* that will be associated with the request: the *system metadata* for the FHIR installation as a whole, the *resource type metadata* for the type of resource being requested (Defn. 6a), the *resource metadata* identified by the data resource request (Defn. 6b), and, the *consent metadata* associated with the data owner. Each *Sec object*'s requirements constructed through the retrieved metadata must be satisfied by one or more credentials sent by the requestor to determine the requestor's trustworthiness. When the controller receives a requestor's trust profile credential and has finished verifying the credential's authenticity, an attempt is made to match it against the $\mathfrak{QHP}$s in each of the four retrieved *Sec objects*. The controller records which of the $\mathfrak{QHP}$s has been satisfied, and creates an SGP based on which $\mathfrak{QHP}$s remain unsatisfied to send back to the requestor. All of the four *Sec objects* must be satisfied for the trust negotiation to be successful. A *Sec object* is satisfied if one of its $\mathfrak{QHP}$ sets is satisfied. A single trust profile credential is potentially capable of satisfying multiple $\mathfrak{QHP}$s across multiple *Sec objects*. During credential exchange, the controller is continually checking the requestor's credentials and matching them to the *Sec objects* until all of the $\mathfrak{QHP}$s are satisfied or the requestor chooses not to send another

credential. If the requestor chooses not to send another credential, the controller checks whether all of the four *Sec objects* are satisfied, executes the release actions, and provides the resource and new trust profile credentials. The controller's final set of release actions are resolved hierarchically from $\mathcal{Sec}_{\mathcal{System}}$ to $\mathcal{Sec}_{\mathcal{Resource}}$ by beginning with the $\mathcal{Sec}_{\mathcal{System}}$ set of release actions and combining with $\mathcal{Sec}_{\mathcal{Resource\ Type}}$ release actions, then $\mathcal{Sec}_{\mathcal{Resource}}$ release actions. When two $\mathcal{RA}$s conflict at different levels, the $\mathcal{RA}$ at the lowest level (closest to the individual resource) takes precedence. The $\mathcal{Sec}_{\mathcal{Consent}}$ release actions are separated from the other three *Sec objects* and are always executed as specified by the data's owner. The *consent* object concerns data owner notifications but may also filter access to the data owner's resources more strictly or release resources more freely to specific organizations and thus override the other three *Sec objects*. Within a single *Sec object*, each $\mathcal{AHP}$ contains a ranking, with higher ranking determining which $\mathcal{RA}$ is executed if there is a conflict between two $\mathcal{RA}$s in two satisfied $\mathcal{AHP}$s.

These last aspects of the framework can be integrated through a front end into FHIR resources and a corresponding HAPI FHIR server implementation. The $\mathcal{Sec}_{\mathcal{System}}$ object is mapped to the FHIR instance, the $Sec_{ResourceType}$ object is mapped to the FHIR resource type (e.g., Observation, MedicationStatement, Procedure, etc.), the $\mathcal{Sec}_{\mathcal{Resource}}$ object is mapped to the individual FHIR resource instance (e.g., a single resource instance of the Observation type), and the $\mathcal{Sec}_{\mathcal{Consent}}$ object is mapped to the Patient whose healtchare data is described by the requested resource. The evaluation of the four *Sec objects* is resolved utilizing the generalized method above, by evaluating the satisfaction of the *Sec objects* starting with the $\mathcal{Sec}_{\mathcal{System}}$ for the FHIR instance, then evaluating the $\mathcal{Sec}_{\mathcal{Resource\ Type}}$ for the resource

type, then the $\mathcal{S}ec_{\mathcal{R}esource}$ for the resource, ending with the $\mathcal{S}ec_{\mathcal{C}onsent}$ object for the patient. The specified release actions for the $\mathcal{QHP}$ set that satisfied the *Sec object* are combined in this order and resolved via security level (FHIR server instance ranking lowest and patient consent ranking highest).

5.2. Controller Configuration

This section describes the configuration of a controller to allow for dynamic creation of *Sec objects* depending on the user's role and resources requested in the Request Context (Defn. 16). Configurations are created utilizing a specialized JSON specification capable of defining multiple dynamic $\mathcal{QHP}$ sets for each role and multiple roles for each *Sec object*. The first part of a JSON schema for configuring a *Sec object* integrated with the FHIR standard is shown in Figure 5.6. Each *Sec object* follows the structure defined in the schema to express the trust profile requirements on a per role basis. The root of the object, shown in the top of Figure 5.6 contains the *Sec object*'s *type* identifier, which can take the values "system" for a $\mathcal{S}ec_{\mathcal{S}ystem}$ object, "type" for a $\mathcal{S}ec_{\mathcal{R}esource\ \mathcal{T}ype}$ object, "resource" for a $\mathcal{S}ec_{\mathcal{R}esource}$ object, or "consent" for a $Sec_{Consent}$ object. The *id* field is dependent on the type and displays the appropriate ID for a System, Resource Type, Resource, or Owner. The ID for a system consists of the domain name of the system (e.g., bmi9.engr.uconn.edu), The ID for a Resource Type is the name of the Resource Type (e.g., Observation, Patient, MedicationStatement, etc.), the ID for a $\mathcal{DR}_{\mathcal{ID}}$ is the unique numeric identifier for the resource, and the ID for an Owner is a unique numeric identifier for the Owner. Example IDs for each *Sec object* type are displayed in Figures 5.2 through 5.4.

```
{ // absence of an instance of type results in the denial of a request of a resource of that type
    "type": "system" | "type" | "resource" | "consent",
    "id": <string>, // system domain | Resource Type | Resource ID | Owner ID
    "role1": [requirement], // same roles as specified in property.role fields
    "role2": [requirement],
    "role n": {requirement}
}
requirement: {
    "affiliation": {property},
    "properties": [property],
    "direct-consent": [direct-consent]
}

property: { // at least one entry must be present, absence of an individual property indicates implicit satisfaction of the property with any value
    "crud": {crud},
    "quantity": <int>, // The number of unique certificates requried whose contents fulfill this property
    "period": {period}, // Specifies time constraints on the timestamp for the certificate type
    "role": Required User role code,// Healthcare professional roles located: https://www.hl7.org/fhir/valueset-participant-role.html
    "owner": owner ID | "request_id", // a value of request_id indicates that the owner ID associated with the requested record should be substituted
    "resource-type": The type of resource required,
    "resource-clearance": "U" | "S" | "TS",
    "system-clearance": "U" | "S" | "TS",
    "system-id": <string>, // system domain (url to initiate trust negotiation)
    "release-actions": [release-actions]
}

crud: {
    "create": <boolean>,
    "read": <boolean>,
    "update": <boolean>,
    "delete": <boolean>
}

direct-consent: {
    "crud": {crud},
    "issuer": <string>, // String representation of distinguished name
    "serial": <int>,
    "public-key": {public-key}
}

public-key: {
    "algorithm": "rsaEncryption", // String representation of encryption algorithm
    "modulus": <String>, // Hex string encoded public key modulus
    "exponent": <String> // String representation of public key exponent
}
```

Figure 5.6. JSON Specification for *Sec Object* Configuration Part 1.

The *role 1*, *role 2*, and *role n* fields represent codes for the roles required for satisfaction of the *Sec object*. Each role is provided an identifier allowing easy extraction of the role from the config. For FHIR integration, the role codes utilize the US Core CareTeam Provider roles (HL7 International, 2019) compiled from the NUCC Health Care Provider Taxonomy Code Set (NUCC, 2020) and SNOMED CT (SNOMED International, 2020). Each role contains a requirement object that encapsulates the requirements for the role. If a role is not specified by the configuration, the *Sec object* cannot be satisfied by that role. The *requirements* field contains three entries, the *affiliation* entry which is an instance of a *property*, the *properties* entry, which is a list of *property* instances, and the *direct-consent* entry, which is a list of *direct-consent* instances. The *affiliation* entry allows the specification of the requirements for an $\mathfrak{AC}_{\mathfrak{Affiliation}}$ certificate through the *property* object. The *properties* field allows the specification of an array of sets of $\mathfrak{ACPs}$, the satisfaction of

one being sufficient for satisfying the *Sec object*. The *direct-consent* supports the addition of a trust profile identity certificate, whose proven owner automatically satisfies the requirements of the *Sec object*.

The *property* field specifies the $\mathcal{PHP}_s$ necessary for satisfaction of the *Sec object*, as well as a meta-property *quantity* that specifies the number of trust profile entries that must have those $\mathcal{PHP}_s$. The *crud* field specifies the CRUD operations (create, read, update, delete) that the requestor may perform if that property set is satisfied by the trust profile. The *period* field specifies a range that the trust profile entry's timestamp must fall within to satisfy the *Sec object*. The *period* field allows for expression as a date range or a relative time, either from a specified time until the present or before a specified time. The *role* field specifies the role that the requestor must have assumed at the time of access and is specified as a role code from the US Core CareTeam Provider roles. The *owner* field specifies the owner of the data in the noted access was, either identified explicitly by ID, or it can have the value "request_id", in which case the ID in the trust profile access must be the same as the ID of the owner of the resource specified in the $\mathcal{RC}$. The *resource-type* field specifies the type of resource the trust profile entry must describe. For FHIR integration, this *resource-type* would describe a FHIR resource type, such as an Observation, Patient, or MedicationStatement. *Resource-clearance* and *system-clearance* refer to the clearances of the requested object matched with clearances listed in $\mathcal{AC}_{\mathcal{DataResourceConfidentiality}}$ and $\mathcal{AC}_{\mathcal{SystemConfidentiality}}$ certificates. The clearance listed in the trust profile must meet or exceed the clearance listed in the configuration to satisfy *resource-clearance* and *system-clearance*. The *system-id* specifies an ID in the same format as the $\mathcal{Sec}_{\mathcal{System}}$ ID (trust negotiation endpoint URL). The

release-actions field specifies a list of release actions that are performed if the *Sec object* is satisfied by that $\mathfrak{PKP}$ set.

A *direct-consent* object provides the fields necessary for uniquely identifying and verifying the ownership of an individual trust profile identity certificate. It contains a *crud* property for specifying the CRUD operations a requestor is allowed to perform if the *direct-consent* object is satisfied. The trust profile identity certificate is uniquely identified through the *issuer* field and the *serial* field. The issuer must be the string representation of the identity certificate issuer's distinguished name, in the order it appears in the certificate. The *public-key* property defines a *public-key* object that has the properties necessary to parse the recorded public key. The displayed public key must use the "rsaEncryption" algorithm, though support for additional encryption algorithms is possible.

Figure 5.7 shows the remainder of the JSON specification that supports the fields in Figure 5.6. The *release-actions* field provides a *log* object specifying the level of logging depending on the level of risk of releasing the requested data, which is dependent on the amount of trust the requestor's trust profile credentials create. A set of strict requirements that generates a large amount of trust would create a low amount of risk, allowing the transaction to be logged as a low risk, specified with the "low" value for the *log* object. The *audit* field specifies whether the transaction must be accompanied by a notification to the local auditor, which carries either a string value representing the auditor's email, or a null indicating that an immediate audit is unnecessary. The *redact* and *modify* fields are a list of *mod-pair* objects that describe how the returned resource should be redacted or modified by specifying individual fields and their new values. The *add* field specifies a list of *download-urls* the requestor utilizes to download the required additional data

automatically through the client implementation. The *period* fields specify the period that the timestamp listed in the trust profile for a data access entry must fall within. Specifying both *from* and *to* fields with a *date* causes the period to evalutate the timestamp as needing to fall between the dates listed in the *to* and *from* fields. If the *from* field is null, the timestamp in the presented trust profile credential must be before the *date* listed in the *to* field inclusive. If the *to* field is null, the timestamp must be after the date listed in *from* up to the present time. Both fields cannot be null or the *period* is invalid. A date object contains a *type* field which specifies if the date is to be interpreted as "literal", "relative", or "current". A type of "literal" indicates that the *value* field, specified in yyyy-MM-dd format, should be parsed as a yyyy-MM-dd date. A type of "relative" indicates that the *value* field should be an int, indicating that the date should be interpreted as being *value* units in the past, where the units are specified by the *unit* field, which may take on the values specified in *time-value*. A date type of *current* indicates that the *Sec object* should utilize the current time at the time of *Sec object* creation.

```
release-actions: {
    "log": "low" | "mid" | "high", // log level
    "audit": <string> | null, // auditor email or null, null indicates that an audit is not required and cancels out a higher level requirement
    "redact": [mod-pair],
    "add": [download-urls],
    "modify": [mod-pair]
}

mod-pair: {
    "name": field to modify,
    "value": the replacement field
}

period: { // both values cannot be null, if both are specified the period must be in between the "from" value and "to" value inclusive
    "from": date | null, // if null, the property is valid if the credential's timestamp is a date from any point in the past up to the "to" value inclusive
    "to": date | null // if null, the property is valid if the credential's timestamp is a date from the "from" value to the time it is now inclusive
}

/**
 * Specify a date in either literal, relative, or current format. A literal format specifies the date in yyyy-MM-dd format.
 * Ex. "2020-03-24" for March 24, 2020.
 * Literal format requires the type and value properties.

 * A relative format specifies a sliding window that is a certain number of units in the past from the current time UTC.
 * Ex. value: 3, unit: year indicates that the date's value should be three years before the current time on the server.
 *     If the current date is 2020-03-24, when this config is parsed, the trust negotiation server should replace
 *     the date value with 2017-03-24. This allows the specification for a credential's timestamp being within three
 *     years of the current time without the need to continuously update the JSON config.
 * Relative format requires the type, value, and unit properties
 *
 * Current format specifies that the controller should use the current time as a date.
 * Current format requires the type property.
 */
date: {
    "type": "literal" | "relative" | "current",
    "value": "yyyy-MM-dd" | <int>, // "yyyy-MM-dd" string format for literal type, int for relative type
    "unit": <time-value> // present if type = relative
}

time-value: "year" | "month" | "day"
```

Figure 5.7. JSON Specification for *Sec Object* Configuration Part 2.

Chapter 6
CT^2 Prototype

This chapter describes in depth the integration of trust profiles and trust negotiation into the Connecticut Concussion Tracker CT^2 app. Trust negotiation support for the CT^2 app is implemented through the combination of four components. The first component is the modified CT^2 mobile Android app, written in Java and supported by the Bouncy Castle cryptographic library, that allows for authorization to individual student concussion records through the trust profile. The second component is the Trust Negotiation Certificate Manager and GUI, which was written in Java, that manages and creates certificates and a certificate tree for testing, where the root consists of self-signed medical authorities, the internal nodes consist of CA certificates for healthcare organizations, and the leaves consist of identity certificates and attribute certificates that encode trust profile data. The third component is a trust profile supporting trust negotiation controller written in Java that: manages the connection with the CT^2 app; oversees the validation of the trust profile certificates and their certificate chains; and, generates new trust profile credentials for a successful trust negotiation. The fourth component is a back-end certificate matching component that: parses the *Sec object* configuration; receives validated trust profile credentials; matches the validated credentials to the *Sec object* requirements; and, produces SGPs when the *Sec objects* have not been fully satisfied. These four components work with the existing, unmodified CT^2 back-end infrastructure (database and RESTful API).

The remainder of this chapter consists of 3 sections. Section 6.1 discusses the CT^2 app and the modifications made to support trust profiles and trust negotiation. Section 6.2

shows the certificate organizer utilized to generate the initial trust profile credentials for the users. Section 6.3 concludes the chapter with a detailed description of the controller. All of the concepts discussed in Sections 6.1 to 6.3 support expected Contribution C: Dynamically Generated Adaptive Access Control Policies and Contribution D: Trust Negotiation Development Framework.

6.1. Modified CT^2 App

The CT^2 app discussed in Section 2.3, shown in Figure 6.1 has been modified to support trust profiles and trust negotiation when viewing a student's concussion data. The initial trust profile certificates are located on the device and accessible to the CT^2 app. The trust negotiation prototype app allows the viewing of a listing of students with concussion data in the remote concussion database. The CT^2 app has an architecture that utilizes a RESTful API as the back-end, as shown in Figure 6.2 which has been adapted from (Rivera Sánchez, A Configurable Framework for RBAC, MAC, and DAC for Mobile Applications, 2017). Notice that the architecture is back ended by the concussion database which is used to store the information on concussions in the bottom of Figure 6.2. The trust negotiation process will involve providing access to the concussion resources based on the trust profile of individual users.

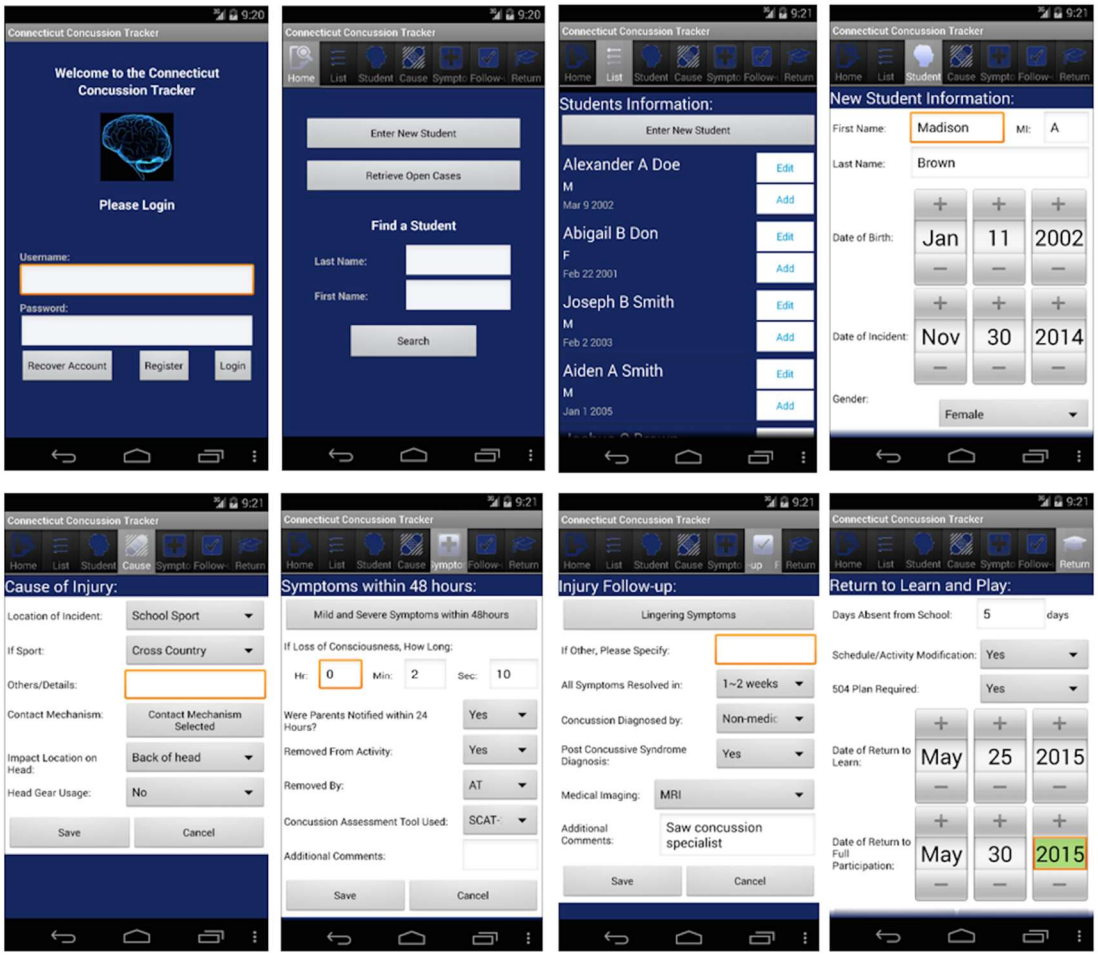

Figure 6.1. The CT^2 Application Screens.

The user can retrieve the open concussion cases or search for a student by first and last name as shown in Screen 2 over the first row in Figure 6.1. The results list, shown in Screen 3 of the first row in Figure 6.1, is then used to select the student whose concussion data the user wishes to view. Accessing an individual concussion case will require the ability to access a subset of the concussion resources, which correspond to the data resources $\mathcal{DR}$ in our model. When the user selects the student's name, the app prompts with a notification that trust negotiation will be required and displays the screen shown in

Figure 6.3. The app automatically creates a Request Context $\mathcal{RC}$ by adding the role of the user's CT^2 account and displays a list of trust profile certificates for the user to add to the trust profile tp_{DW}, with the Data Resource $\mathcal{DR}$ already filled out automatically by the app based on the student's concussion data previously chosen by the user. In this instance, the resource in the $\mathcal{DR}$ is the concussion record for the previously chosen student. Pressing the send button after choosing the trust profile certificates sends the $\mathcal{RC}$ to the Trust Negotiation Controller, to be detailed in Section 6.3.

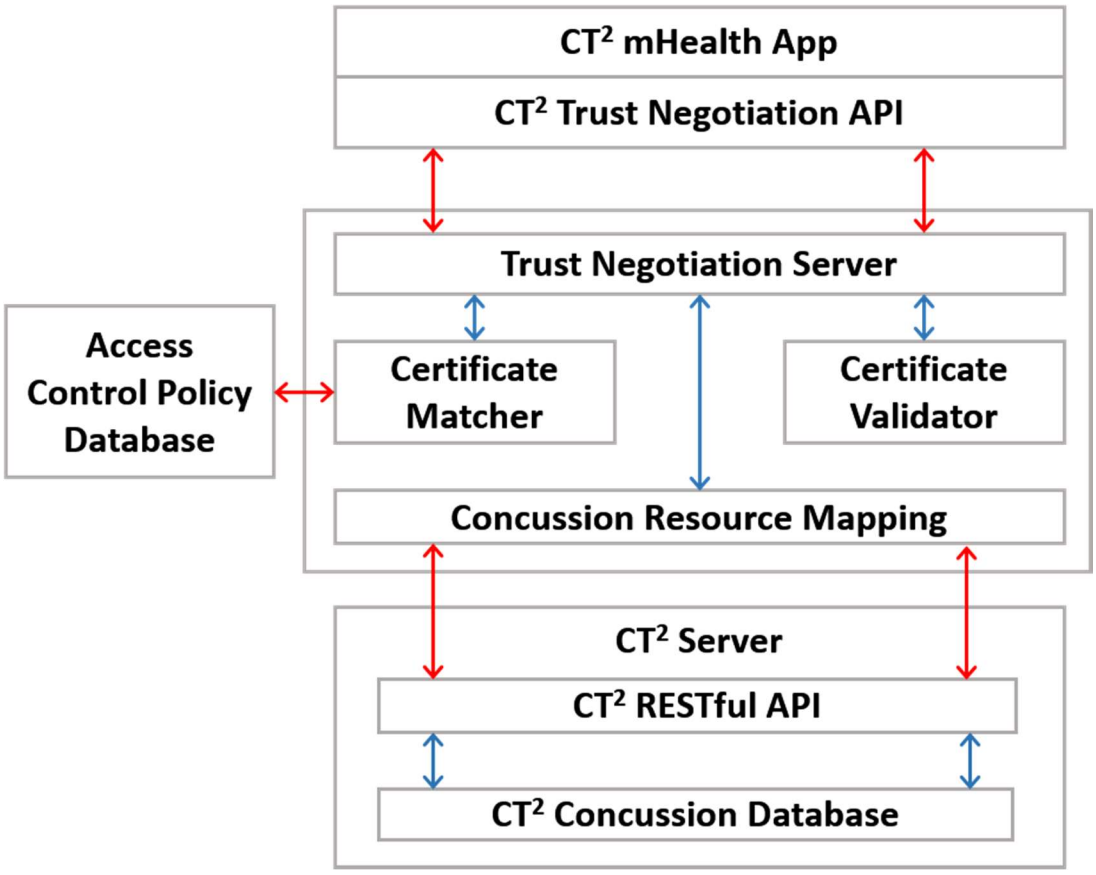

Figure 6.2. The CT^2 Architecture.

The Trust Negotiation Controller receives the $\mathcal{RC}$ from the app, processes as described in Section 6.3, and determines whether the certificates chosen by the user are sufficient to obtain access to the requested concussion data. If the tp_{DW} from the $\mathcal{RC}$ is insufficient, the CT^2 app receives an SGP from the controller. This SGP is displayed in Figure 6.4. The SGP provides a description to the user of the remaining unsatisfied requirements of the controller's generated *Sec objects* allowing the user to choose trust profile credentials that fulfill the controller's requirements. Once the user chooses the trust profile credentials and sends them to the controller, the process continues until the trust negotiation controller's trust profile requirements are satisfied. Once the presented trust profile credentials are sufficient to satisfy the controller's trust profile requirements, the controller forwards new certificates detailing access to the student's concussion data. These trust profile credentials are automatically added to the user's trust profile credentials located on the device and will appear in the user's trust profile credentials listing, allowing those credentials to be utilized in future trust negotiation attempts. On a successful trust negotiation, the requested concussion data is sent to the CT^2 app, which can then display the requested concussion data for the student as shown in Figure 6.5. If the user cannot supply sufficient credentials to obtain access to the requested concussion data, the connection to the trust negotiation server is closed and the app displays a failure message to the user, shown in Figure 6.6.

Figure 6.3. The Build Request Context Screen.

Figure 6.4. The Server Governance Policy Screen.

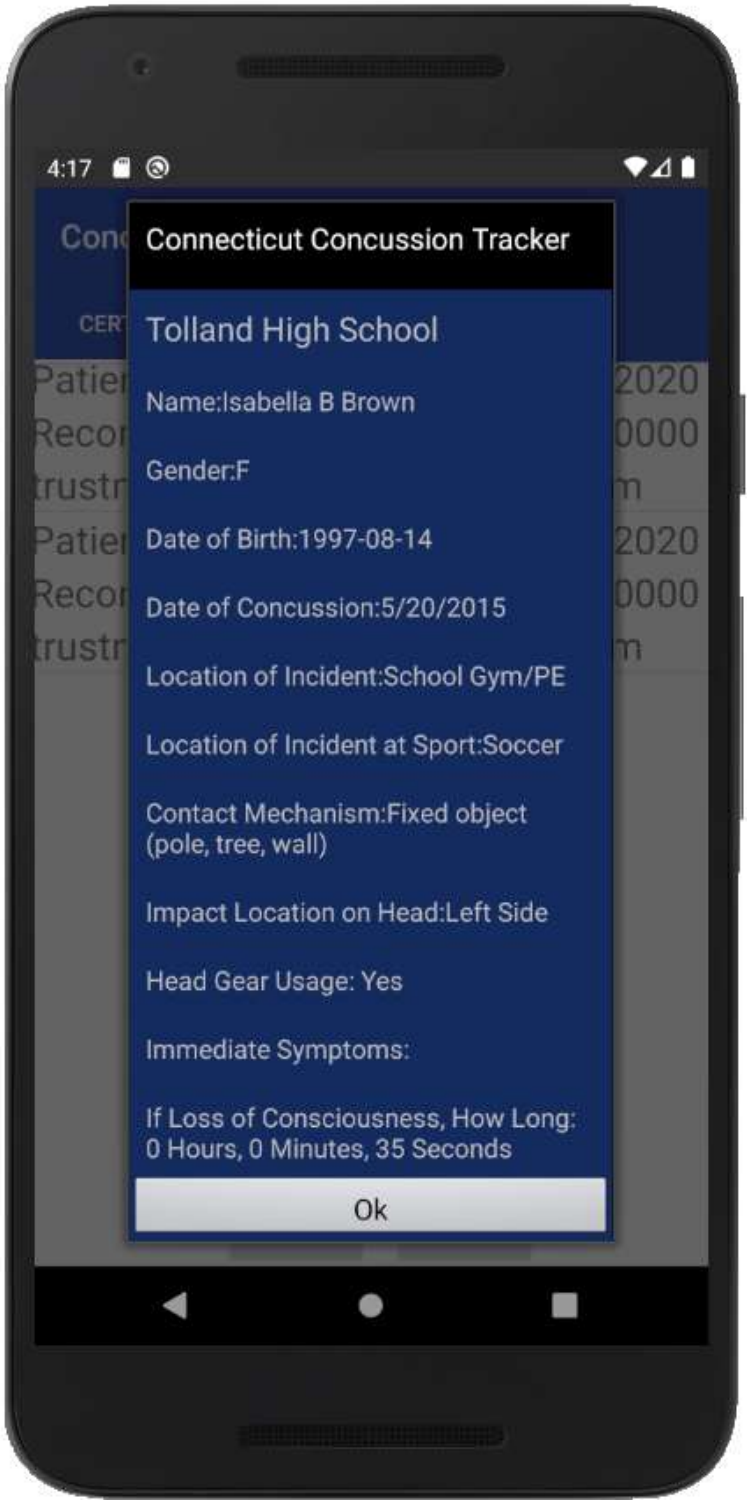

Figure 6.5. Concussion Data Received.

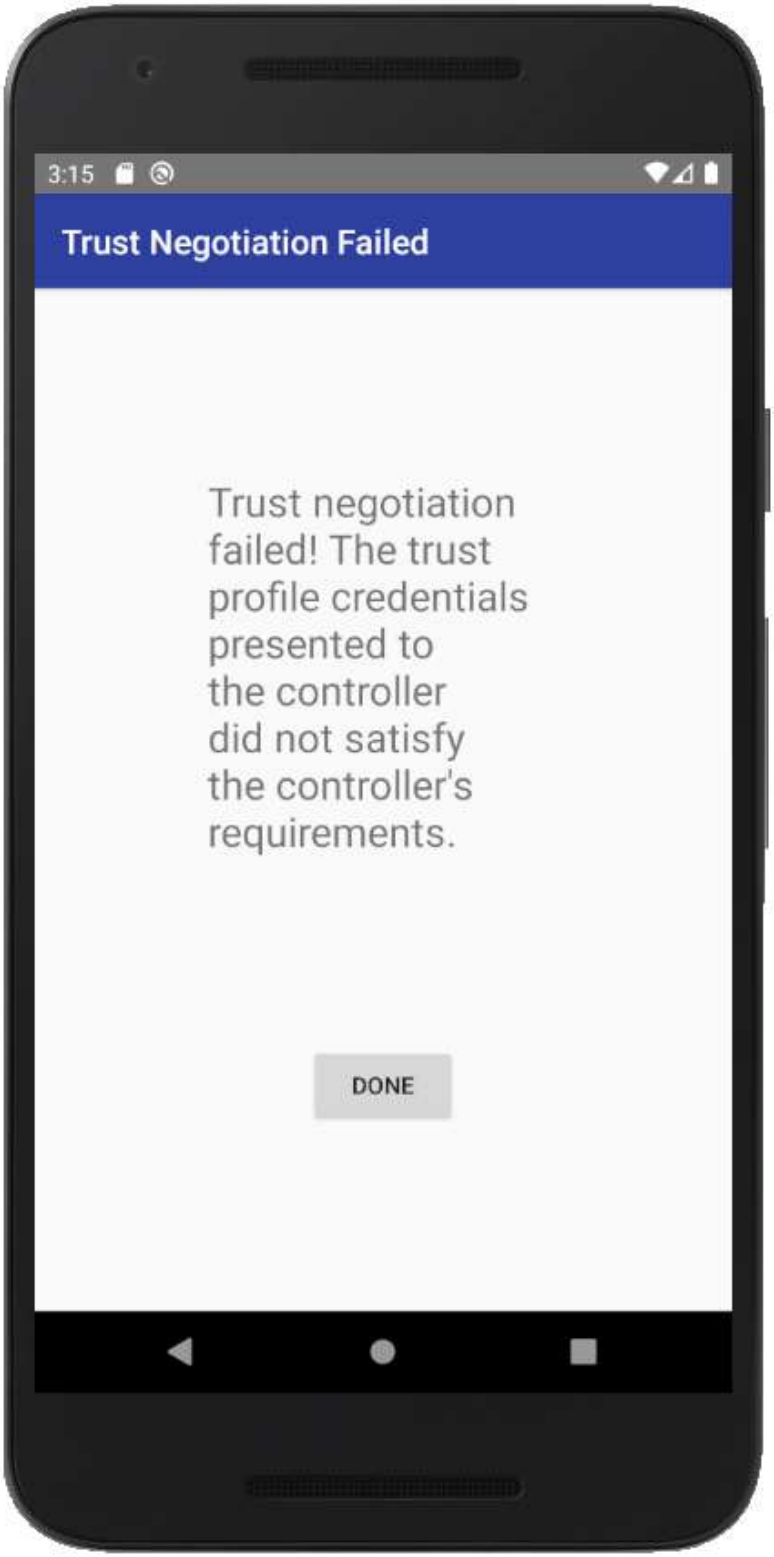

Figure 6.6. The Trust Negotiation Failure Screen.

6.2. Trust Negotiation Certificate Manager

The Trust Negotiation Certificate Manager GUI is shown in Figure 6.7. The Certificate Manager contains three main views: the *Certificate Authorities View*, which shows all organizations capable of signing a valid certificate; the *Identity Certificates View*, which lists all X.509 identity certificates belonging to a user and signed by the highlighted certificate in the Certificate Authorities View; and, the *Attribute Certificates View*, which lists all the Attribute Certificates belonging to the user whose identity certificate is selected in the Identity Certificates View. The manager is configured to check a configured folder and its subfolders for the presence of certificates and organizes all of the certificates into an interal tree structure starting at the root, self-signed certificates down to the attribute certificates, where the parent of each certificate is the certificate that signed it. Attribute certificates are attached as children to the identity certificate they are attached to by issuer/serial number combination.

The Certificate Authorities' View displays all of the certificate authority certificates organized into a tree view, where the child certificates in the lower branches of the tree are signed by its parent certificate. All of the certificates within this view represent an authority like the medical authority or an organization that participates in trust profile based trust negotiation. When a certificate is selected in the Certificate Authorities View, the subject common name, issuer common name, and serial number of the certificate appear in the fields directly below the Certificate Authorities View. Certificate authority certificates are created through OpenSSL (OpenSSL Software Foundation, 2018) and must be added to the folder that the GUI is configured to search.

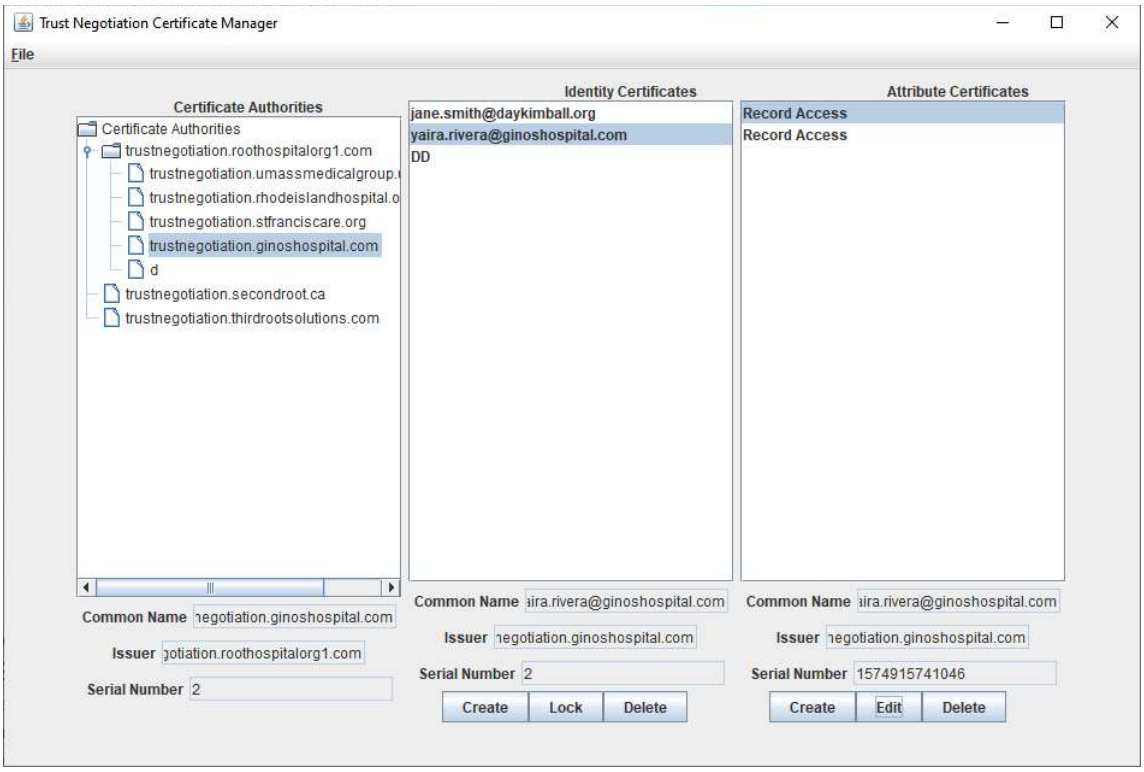

Figure 6.7. The Trust Negotiation Certificate Manager Interface.

The Identity Certificates View contains a listing of all of the identity certificates that were signed by the selected certificate in the Certificate Authorities View. These identity certificates are listed by the common name of the subject's distinguished name field. When an identity certificate is selected, the Identity Certificates View lists the identity certificate's subject common name, the identity certificate's issuer common name, and the identity certificate's serial number below the Identity certificate listing. The button panel at the bottom of the Identity Certificates' View provides Create, Lock, and Delete buttons, adding functionality for creating and deleting identity certificates, as well as featuring a Lock on the Identity Certificate selection. The Lock functionality is utilized in Attribute Certificate selection and will be detailed during the Attribute Certificates View description. When the create button is pressed and the user has previously selected a

certificate in the Certificate Authorities View, a prompt appears, shown in Figure 6.8, that displays the distinguished name of the issuer and provides text fields to enter the distinguished name of the subject. Once the Create button in the prompt is pressed, the Certificate Manager retrieves the private key of the selected certificate authority certificate, signs a new identity certificate, and places it in the signed folder of the selected certificate authority. The delete button removes the certificate file from the file system.

The Attribute Certificates View displays a list of all attribute certificates belonging to the selected certificate in the Identity Certificates View. When an Attribute Certificate is selected, the common name of the parent identity certificate, the common name of the attribute certificate issuer, and the serial number of the attribute certificate are displayed in the text fields below the attribute certificate listing. The buttons below the text fields feature Create, Edit, and Delete corresponding respectively with attribute certificate creation, editing, and deletion. Note that since the attribute certificate is cryptographically signed, editing results in deleting the old attribute certificate and creating a new attribute certificate with the given values. The edit screen is shown in Figure 6.9. If the ASN.1 button is pressed, the screen in Figure 6.10 appears showing the certificate in ASN.1 encoded form, which can be copied into an ASN.1 decoder to verify the contents of the attribute certificate. When the create button is pressed, a prompt appears with a dropdown for specifying which type of $\mathfrak{AC}$ (Defns. 14a, 14b, 14c, and 14d in Chapter 4, Section 4.2) to create. When selected, a prompt appears with the relevant, editable fields for that type of certificate. Confirming the data results in the Certificate Manager creating and saving a new attribute certificate, whose parent is the highlighted identity certificate in the Identity

Certificates View, and whose issuer is the selected certificate in the Certificate Authorities View.

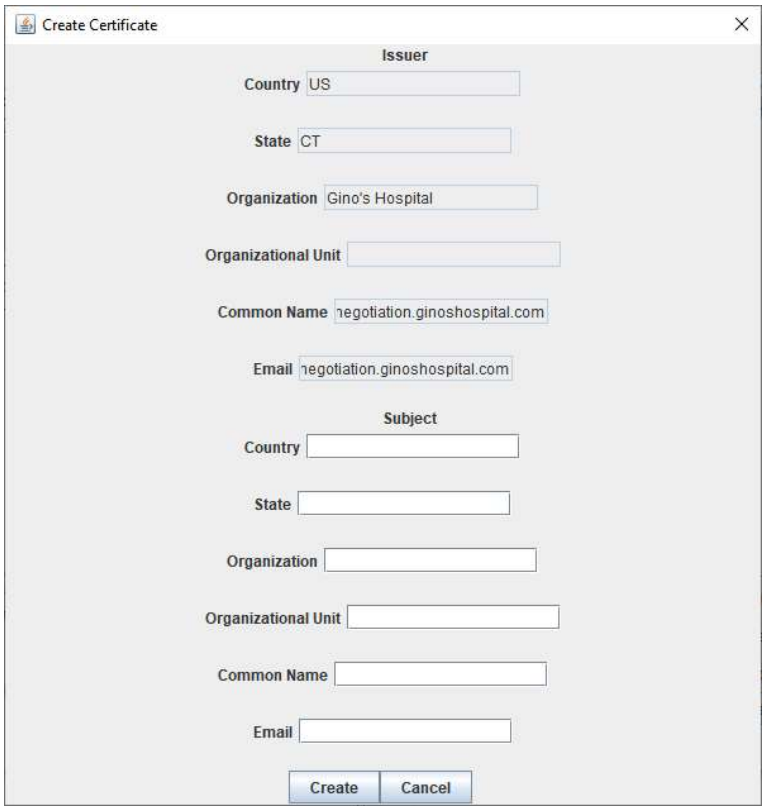

Figure 6.8. The Create Identity Certificate Interface.

To improve usability and ease of finding and reading certificates, the view of the certificate listing to the right automatically updates with a listing of child certificates when a certificate is selected in the certificate view to the left. For example, when a certificate is selected in the Certificate Authorities View, the Identity Certificates View is updated with a listing of all of the identity certificates signed by that certificate. Additionally, when an identity certificate is selected, the Attribute Certificates View is updated with a list of all

attribute certificates attached to the identity certificate. In order to simulate a controller with separate attribute authorities and certificate authorities, as is required by Section 4.5 of RFC 5755 (Farrell, S et al., 2010), the Lock button functionality of the Identity Certificate freezes the Identity Certificate View, allowing a certificate from the Certificate Authority View to be selected without the current identity certificate being deselected. This lets a separate attribute authority certificate that hasn't signed the identity certificate to be selected for attribute certificate signing.

Figure 6.9. The Attribute Certificate Edit Screen.

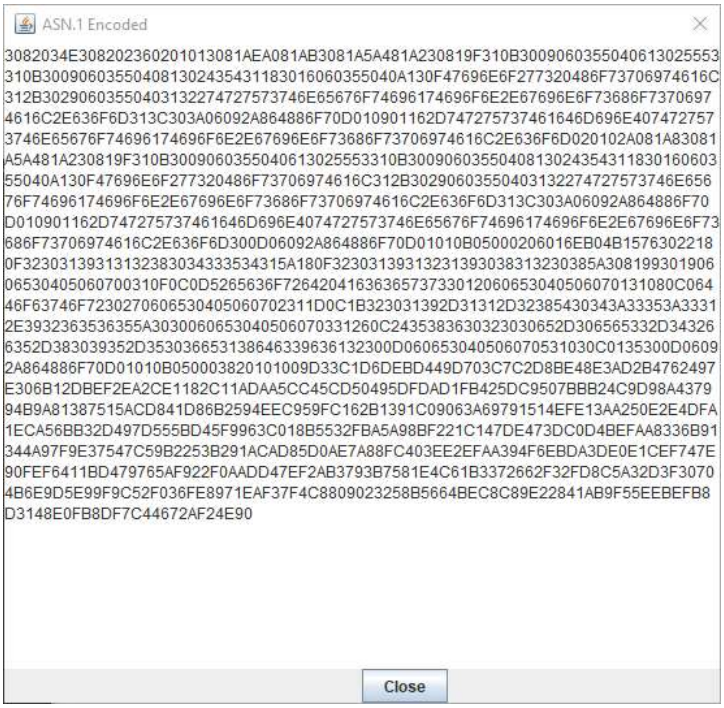

Figure 6.10. The ASN.1 Display.

6.3. Trust Negotiation Controller

The trust negotiation controller is written in Java, utilizing Apache Struts (The Apache Software Foundation, 2018) for the web interface and Bouncy Castle (Legion of the Bouncy Castle Inc., 2013) for certificate parsing and generation, both running on an Apache Tomcat (The Apache Software Foundation, 2020) server. The trust negotiation controller consists of two components. The first is a component that: manages the connection with the modified CT^2 app; negotiates the release of credentials; validates the certificates presented by the requestor (X.509 digital signature inspection, certificate chain validation, and certificate revocation list checking); and, generates new credentials if trust negotiation is successful. The second component receives validated certificate chains, parses the *Sec object* JSON configurations from Chapter 5.2, generates a *Sec object* for

each of the four security levels (system, resource type, resource, and consent), and attempts to match the attributes for each shared certificate with the requirements it parses from the *Sec object* configurations.

The first component operates a Struts server that receives the initial request and request context $\mathcal{RC}$ from the modified CT^2 app via the RESTful Trust Negotiation API. The request is directed at the struts server's /trustnegotiation/ endpoint. When the initial concussion request context is received, the requestor's role, the Resource Type, the Resource ID, and the Patient ID are extracted. The $\mathcal{RC}$ is extracted and the trust negotiation server retrieves the user's requested role, the $\mathcal{DR}$, and the tp_{DW} through its certificate upload URL. The trust negotiation server checks to make sure that the resource at the URL matches the requested resource in the $\mathcal{RC}$ and performs validation on the certificates in the tp_{DW} utilizing the process discussed in Chapter 2.1 following the X.509 standard's description of certificate validation. The resource request and any uploaded attribute certificates are then passed to the second component to begin matching the trust profile credentials encoded in the certificates to the security requirements in the *Sec object* configuration.

The second component receives the requested role, Resource Type, Resource ID, and Patient ID and makes a request to the local MySQL database housing the *Sec object* configurations for the configurations needed for the current request. The $\mathcal{Sec}_{\mathcal{System}}$ configuration is retrieved utilizing the local S_{ID}, in this case bmi9.engr.uconn.edu, $\mathcal{Sec}_{\mathcal{Resource\ Type}}$ noted in the $\mathcal{RC}$, the $\mathcal{Sec}_{\mathcal{Resource}}$ noted in the $\mathcal{RC}$, and $\mathcal{Sec}_{\mathcal{Consent}}$ is retrieved from the database from the Patient ID in the $\mathcal{RC}$. A new *Sec object* Java instance is created for each *Sec* configuration with the JSON configuration utilized to communicate the trust profile entries

that must be present to satisfy a *Sec object*. When it receives new certificates, the certificates are placed into a tree structure that groups the certificates based on the System $\mathcal{S}$ that the access took place on and by the actual access that occurred. The new certificates are matched against each of the $\mathcal{QKP}$ sets in each of the *Sec object* instances, being added to a set of certificates that records when an access for sensitive data satisfies the properties of a *Sec object*. After the new certificates have been processed, each *Sec object* is queried to produce an SGP if the certificates presented thus far have not satisfied its requirements. The resulting SGPs are returned to the first component, which returns the SGP to the CT^2 app, resulting in a prompt to the requestor to produce trust profile credentials that satisfy the SGPs.

This process of communication and credential exchange between the CT^2 app and the trust negotiation controller continues until each of the *Sec objects* are satisfied or until the requestor chooses to end trust negotiation without obtaining the requested data. When each of the *Sec objects* are queried for SGPs and each return a response indicating that they are fully satisfied, the new attribute certificates describing the access to the requested resource are created. The trust negotiation controller then contacts the concussion app backend through its RESTful API and makes a request for the data originally requested. The release actions for the satisifed $\mathcal{QKP}_s$ are compiled and executed in accordance with Chapter 5.1.5, including logging the transactions at the levels listed in the *Sec objects* compiled through the $\mathcal{RA}_s$. The list of download URLs for the new certificates and the requested data are passed back to the CT^2 app, which automatically downloads the new certificates to the user's trust profile and displays the requested data to the user, as shown in Figure 6.5. The connection between the app and the controller is then terminated.

Chapter 7
Conclusion

This dissertation presented and explained a method of authorization between two entities that have no pre-existing relationship utilizing trust negotiation (Winsborough, Seamons, & Jones, 2000) in conjunction with the trust profile introduced in this dissertation to securely exchange credentials based on the trust profile owner's access history of sensitive healthcare data. A combination of ABAC (Hu, et al., 2014) to allow flexibility in the EHR environment for different combinations of credentials, RBAC (Ferraiolo, Sandhu, Gavrila, Kuhn, & Chandramou, 2001) to properly model current localized methods of access control in an EHR envornment and to inform the trust negotiation controls as to a proper representation of access history for the given role, and MAC (Bell & La Padula, 1976) to implement multi-level security for the protected data were used to regulate access to sensitive data resources. Successful trust negotiation results in a set of new credentials that are passed back to the requestor, who adds those credentials to their personal trust profile and can be utilized in future attempts at trust negotiation. The main objectives have been four-fold: 1. define a set of Infrastructure Requirements to Promote Trust Among Organizations Participating in Trust Negotiation that allows healthcare organizations to trust in credentials issued by each other, even in the event the healthcare organizations do not personally know each other; 2. define an Integrated Trust Profile Model for Recording Complete Records of User Access to Sensitive Data to document a user's sensitive EHR access as a method of building trust with unknown healthcare organizations; 3. define Dynamically Generated Adaptive Access Control Policies to map trust profile credentials

to ABAC, RBAC, and MAC; and, 4. define a Trust Negotiation Development Framework that incorporates these contributions into the FHIR standard and our implementation of the CT^2 app.

The remainder of this conclusion is organized into 3 sections. Section 7.1 summarizes this dissertation while highlighting the four main objectives detailed above. Section 7.2 builds on this by discussing the four research contributions of this dissertation: Contribution A: Infrastructure Requirements to Promote Trust Among Organizations Participating in Trust Negotiation; Contribution B: Integrated Trust Profile Model for Recording Complete Records of User Access to Sensitive Data; Contribution C: Dynamically Generated Adaptive Access Control Policies; and, Contribution D: Trust Negotiation Development Framework. In Section 7.3, we discuss ongoing research and identify future directions for trust profile based trust negotiation including but not limited to: framework extensions that streamline the trust negotiation process, demonstration of our approach in other domains, further integration of the trust negotiation framework with FHIR, improvements to the trust profile requirements specification, framework deployment improvements, and a formal security evaluation of trust profile based trust negotiation.

7.1. Summary

The research presented in this dissertation works to create a trust profile based on a user's access history to sensitive data resources as a new means of obtaining authorization to similar sensitive data resources, without the need for the user or the remote controller to have a pre-existing relationship. The main focus of the dissertation was to create an automated process where a user's history of access to sensitive data resources allows a

remote controller to infer trustworthiness in handling future sensitive data resources by: generating a series of trust profile requirements for the user, communicating the requirements to the user, negotiating the release of trust profile credentials that fulfill the requirements, matching the presented credentials to the requirements, and generating new credentials when the requirements are fulfilled. The discussion of this process was presented throughout six chapters.

Chapter 1 introduced the main research areas and provided a high level overview for enhancement of existing authorization models with trust negotiation and trust profiles. Section 1.1 discussed the motivation for trust profiles and trust negotiation from a healthcare perspective, describing the state of healthcare security and the issues within healthcare that demand new forms of dynamic authorization to sensitive healthcare data. Section 1.2 discussed the motiviation for trust profiles and the potential applications of trust profiling to enable dynamic authorization between two arbitrary parties with no pre-existing relationship. Section 1.3 explained a high level overview of our approach to integrate trust profiles and trust negotiation into an authorization solution. Section 1.4 provided a listing of the research objectives and the expected contributions of this dissertation. Section 1.5 discussed the published works written in support of the research discussed in this dissertation. Section 1.6 concludes the chapter with an outline concerning the structure of the remainder of the dissertation.

Chapter 2 provided necessary background on concepts utilized in the construction of our approach to creating a trust profile based trust negotiation authorization framework throughout the remainder of the dissertation. Section 2.1 discussed the current state of trust and interoperability support between healthcare organizations in support of our later

integration of trust profiles and trust negotiation into the healthcare setting. Section 2.2 described access control models commonly utilized in current authorization techniques and the manner in which we integrate them into our work. Section 2.3 provided an overview of the Fast Healthcare Interoperabilty Resources (FHIR) standard and the HAPI FHIR Java implementation of the standard in support of the proof of concept prototype discussed in Chapter 6. Section 2.4 discussed the Connecticut Concussion Tracker (CT^2) app, a collaboration between the Departments of Physiology and Neurobiology, and Computer Science & Engineering at the University of Connecticut and the Schools of Nursing and Medicine, which provided a testbed for our prototype in Chapter 6.

Chapter 3 presented the infrastructure requirements for trust negotiation that create a network of trust between systems participating in trust negotiation via a set of identity and attribute certificates that encode trust profile data. Section 3.1 began the chapter with a discussion of identity and attribute certificates, their formats, and their useage defined in the X.509 standard. Section 3.2 gave background on the basic trust negotiation process, including: credential exchange, certificate validation, and credential expression generation; and, provides an example of trust negotiation from a healthcare perspective. Section 3.3 introduced the trust profile certificate infrastructure that allows trust profile credentials encoded in the identity and attribute certificates to be shared and trusted among all systems participating in trust negotiation. Section 3.4 described the structure of a controller that receives trust negotiation requests, generates requirements for the user's presented trust profile, guards access to sensitive healthcare data, and generates new trust profile credentials on a successful trust negotiation attempt.

Chapter 4 presented a detailed description of a formal model for trust profiles and integration of trust profiles with adaptive trust negotiation. Section 4.1 introduces the trust profile concept and combines it with the identity and attribute certificate concepts introduced in Section 3.1, as well as introducing the new trust profile credential generation process. Section 4.2 introduces a detailed model and definitions for trust profile structure and controller interaction, including definitions for: the user, controller, and resources; the encoding of entries in the trust profile's X.509 identity and attribute certificates; and, the trust negotiation interactions between the users and controllers during the trust negotiation process. Section 4.3 ties together the trust profile definitions by providing a detailed example utilizing the healthcare field that describes the interactions between the user and controller, the process of utilizing the trust profile to obtain access to sensitive resources, and obtaining new trust profile credentials. Section 4.4 describes related work to trust negotiation and compares it to our new implementation.

Chapter 5 introduced the trust negotiation development framework by providing: a controller infrastructure for applying security metadata to resources at multiple levels, a method for defining trust profile requirements, and a method for combining the multi-level security metadata into an access control decision. Section 5.1 explained the *Sec object* concept that parses a configuration for its level, matches incoming trust profile credentials to the security requirements the configuration represents, and decides whether the presented trust profile credentials are sufficient to allow access to its level of security. Sections 5.1.1 to 5.1.4 introduced the four types of *Sec objects* and the security levels: the *system level* that protects all resources of a system, the *resource type level* that protects all resources of a certain type, the *resource level* that protects an individual resource, and the

consent level that allows the data's owner to set security requirements on the data. Section 5.1.5 provided a method for combining the requirements of the four types of *Sec objects* to decide whether access to the requested resource is allowed. Section 5.2 detailed the JSON schema of a configuration and provided example configurations for the four types of *Sec objects*.

Chapter 6 discussed the prototype trust profile supporting trust negotiation framework incorporated into the existing Connecticut Concussion Tracker CT^2 app. Section 6.1 discussed the modifications made to the CT^2 app to support authorization to concussion data via trust profile based trust negotiation and pictured the app with concussion data and the additional screens that support trust negotiation. Section 6.2 showed the Trust Negotiation Certificate Manager that manages the manual creation of certificates for certificate authorities and a user's initial trust profile certificates. Section 6.3 detailed the Trust Negotiation Controller that: provides the communication point for trust negotiation between the CT^2 app and the concussion data, validates certificates, provides the *Sec object* creation and credential matching capabilities, decides whether access to the data is allowed, and generates new trust profile certificates and retrieves the requested data on a successful trust negotiation attempt.

7.2. Research Contributions

This section revisits the expected research contributions presented in Section 1.4 of Chapter 1 and describes how each was attained throughout the chapters of this dissertation. The trust profile based trust negotiation infrastructure and framework has the following contributions:

A. **Infrastructure Requirements to Promote Trust Among Organizations Participating in Trust Negotiation:** This contribution specified a structure for disseminating trust among the trust negotiation participants, displayed in the left side of Figure 1.1 of Chapter 1 and represented by the Trust Building process in Figure 1.2 of Chapter 1. This contribution allows trust in the trust profile certificates to be established through the given application of the X.509 standard, allowing the utilization of a decentralized network structure, increasing fault tolerance and efficiency. Chapter 3 supports this contribution by providing a network structure (shown in Figure 3.3), controller structure (shown in Figure 3.4), and providing a detailed description of the trust profile certificate infrastructure (Section 3.3).

B. **Integrated Trust Profile Model for Recording Complete Records of User Access to Sensitive Data:** This contribution presented a formal model for the completion of a trust negotiation process utilizing trust profile credentials, displayed in the Trust Negotiation level of Figure 1.2 and represented in Figure 1.1 by the Identity Certificates, Attribute Certificates, and Trust Profile on the left side and the process of exchange between the User Trust Agent and the Controller Trust Agent. Chapters 3 and 4 supported this contribution. Chapter 3 supported this contribution through an overview of the trust negotiation process and description of the controller structure. Chapter 4 supported this contribution by providing formal defninitions for the participants in trust profile based trust negotiation

(Defns. 1-3 in Section 4.2 of Chapter 4), the structure of trust profiles (Defns. 8-15 in Section 4.2 of Chapter 4), and the interactions of the requestor and controller (Defns. 4-7, 16-21 in Section 4.2 of Chapter 4). This allowed the creation of a trust profile and methods for negotiating the release of sensitive data.

C. **Dynamically Generated Adaptive Access Control Policies:** This contribution incorporates access control models into the trust negotiation process by allowing the requested resources to be annotated with security metadata and allowing the controller to utilize that security metadata to make a decision regarding access, represented in Figure 1.2 by the Security Policies level and the Access Control Policies on the right side of Figure 1.1. Chapters 3, 4, and 5 supported this contribution by defining the integration of access control policies into attribute certificates, the trust profile, and the controller respectively. Chapter 6 supported this contribution by describing a controller implementation that supports the dynamic access control policies. This contribution allows the integration of common access control models into trust negotiation.

D. **Trust Negotiation Development Framework:** This contribution defines a process for combining Contributions A, B, and C into a coherent whole that provides a unified framework capable of performing trust based trust negotiation. Chapters 5 and 6 supported this contribution by creating a controller definition that dynamically generates access control policies and by detailing a prototype implementation of the research presented in this

dissertation. This contribution allows the easy deployment of a trust negotiation network that incorporates trust profiles as credentials.

7.3. Ongoing and Future Work

This dissertation presents research that has potential for future improvements and extensions to our work on adaptive trust negotiation. A list of ongoing and future topics includes: an extension to the framework in order to model batch processing of resource requests and the ability to delegate trust profile credentials; demonstrating the adaptive, trust profile based trust negotiation framework in other domains; exploring the integration of the trust negotiation framework concepts directly into FHIR resources by leveraging the profile extension; improving communication of requirements and dissemination of credentials through the improvement of the configuration specification, SGPs, and providing automatic fulfillment of the provided SGP by the requestor's trust agent; improving the trust negotiation framework's deployment by streamlining the trust negotiation framework's setup through standardized trust profile credential storage and docker container deployment; and, performing a formal security evaluation of the presented trust profile based trust negotiation framework.

Framework Extensions: The framework currently does not efficiently handle transactions that require the retrieval of multiple resources in a batch format. This is apparent for students that may have multiple concussion records in the prototype, or if trust negotiation is enabled, to retrieve the initial listing of students as under FHIR each student would be listed in a separate Patient resource returned as a bundle. Future work could result in a defined method for resolving a trust negotiation attempt for multiple resources of

differing types in one transaction. Delegation of trust profile entries is another feature that could be added to the framework to enable secure delegation of trust profile credentials, which would allow others to assist in the requesting of sensitive data (e.g., an assistant retrieving healthcare records a physician will need for an appointment). Future work could implement this by creating a method for temporary delegation of a subset of a user's trust profile to another user, allowing a user to securely obtain access to data on behalf of another.

Demonstration of Trust Profiles, Sec Objects, and Controllers in Other Domains: Trust negotiation and trust profiles have applications to domains other than healthcare where a dynamic coalition of multiple specialists have an unforseen need to share secure data. The dynamic sharing of sensitive criminal investigation data among multiple jurisdictions in the legal community could lead to increased cooperation among law enforcement and increased arrests when crimes are committed in multiple jurisdictions. The financial industry is also an applicable domain, as shown by an online purchase trust negotiation implementation (Ryutov, Zhou, Neuman, Leithead, & Seamons, 2005) that utilizes transaction history. The future work considers a methodology for the adaptation of trust profiles and trust negotiations to any domain.

Integration of Trust Negotiation Framework with FHIR: The *Sec object* configurations in the prototype are hosted in a separate database instance and are retrieved through concussion identifiers by the controller when the request is first received. The profile extension is a potential method for embedding a resource's requirements directly into the resource itself. The resources would be able to retrieve multiple *Sec objects* in one request since the resources know their system, resource type, and their patient. This would

also allow easy updating of the *Sec objects* and through the FHIR interface. Additionally, the Security Labels through extensions are another resource provided by FHIR that allows the embedding of access control data, which has the potential to provide additional security data and the potential enhancement of the Release Actions.

Communication of Requirements: While versatile and expressive, the current configuration schema for representing *Sec object* requirements could be improved to reduce redundancy when multiple AHP property sets share similar but not completely overlapping requirements. The improvement could take the form of a property set specified in the properties array itself behaving as a tree where the nodes closer to the roots represent common requirements among its subbranches, and their children represent unique divergences in the trust profile requirements relative to their parents. Additionally, the SGP requirements could be streamlined in the same manner and described to the user in a more intuitive display than the example provided in Chapter 6 in Figure 6.4, where the SGP JSON is printed directly to the screen. Finally, since a trust profile can grow arbitrarily large over the course of a user's entire career, the app could automatically retrieve the proper credentials to fulfill the SGP, only asking the user for confirmation before sending the credentials back to the controller.

Trust Negotiation Framework Deployment Improvements: The deployment of the Trust Negotiation Framework could be improved by making a configurable docker container available that can be run as an easy method for adding trust negotiation to a server housing sensitive data. Through proper docker configuration, a new trust negotiation URL could be implemented that can be connected to through a RESTful API, where the controller is configured to have access to the underlying sensitive data. Additionally, the

trust profile deployment to mobile devices could be improved by properly handling cases where multiple people may need to share the same device, such as in a healthcare setting where separate, security hardened devices may be shared among healthcare employees. In this case, it must be easy for the user to retrieve and use the private keys of their own trust profile, while making it impossible for a user to access the private keys of another user's trust profile.

Formal Security Evaluation: The current implementation of trust profile based trust negotiation lacks a formal security evaluation of its parts and operation that is capable of providing a list of guarantees regarding the usage of trust profiles in both successful and unsuccessful situations. Performing a formal security evaluation of the presented trust profile based trust negotiation framework, as well as the presented CT^2 app prototype, would offer data on possible vulnerabilities that could occur in a potential full deployment of a trust profile based trust negotiation implementation over a large network. This information is useful to medical authorities, whose expertise is necessary to build trust among all of the participants in a healthcare-focused trust negotiation scheme.

References

Alhaqbani, B., & C., F. (2008). Access Control Requirements for Processing Electronic Health Records. In A. ter Hofstede, B. Benatallah, & H.-Y. Paik, *Business Process Management Workshops* (pp. 371-382). Berlin, Heidelberg: Springer Berlin Heidelber.

Bell, D. E., & La Padula, L. J. (1976). *Secure Computer Systems: Unified Exposition and Multics Interpretation.* Bedford, Mass.: MITRE Corp.

Centers for Disease Control and Prevention. (2018, 9 24). *Introduction | Meaningful Use | CDC*. Retrieved from https://www.cdc.gov/ehrmeaningfuluse/introduction.html

Cisco. (2020, March 9). *Cisco Annual Internet Report (2018-2023*. Retrieved May 20, 2020, from https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html

Connecticut General Assembly. (2015). *Substitute for Raised H.B. No. 6722*. Retrieved from https://www.cga.ct.gov/asp/cgabillstatus/cgabillstatus.asp?selBillType=Bill&which_year=2015&bill_num=6722

Cooper, et al. (2008, May). *Internet X.509 Public Key Infrastructure Certificate.* Retrieved from https://tools.ietf.org/html/rfc5280

Demurjian, S., Sanzi, E., Agresta, T., & Yasnoff, W. (January-June 2019). Multi-Level Security in Healthcare using a Lattice-Based Access Control Model. *IGI International Journal of Privacy and Health Information Management (IJPHIM), 7*(1), 80-102.

Elkhodr, M., Shahrestani, S., & Cheung, H. (2011). Enhancing the security of mobile health monitoring systems through trust negotiations. *Local Computer Networks (LCN), 2011 IEEE 36th Converence on* (pp. 754-757). Bonn: IEEE.

Epic Systems Corporation. (2020, January 4). *Epic*. Retrieved from https://www.epic.com/

Facebook. (2020). *Facebook Login.* Retrieved May 21, 2020, from Facebook: https://developers.facebook.com/docs/facebook-login/web/

Farrell, S et al. (2010, January). *An Internet Attribute Certificate Profile for Authorization*. Retrieved July 5, 2020, from The Internet Engineering Task Force (IETF®): https://tools.ietf.org/html/rfc5755

Farrell, S., & Housley, R. (2002, April). *An Internet Attribute Certificate Profile for Authorization*. Retrieved from The Internet Engineering Task Force (IETF®): https://www.ietf.org/rfc/rfc3281.txt

Fernández-Alemán, J., Señor, I., Lozoya, P., & Toval, A. (2013). Security and privacy in electronic health records: A systematic literature review. *J Biomed. Inform, 46*(3), 541-562.

Ferraiolo, D. F., Sandhu, R., Gavrila, S., Kuhn, D. R., & Chandramou, R. (2001). Proposed NIST standard for role-based access control. *ACM Transactions on Information and System Security (TISSEC), 4*(3), 224–274.

Gartner. (March 19, 2015). *Gartner Says Global Devices Shipments to Grow 2.8 Percent in 2015.* Egham, UK: Gartner. Retrieved from http://www.gartner.com/newsroom/id/3010017

Google. (2020, May 20). *Using OAuth 2.0 to Access Google APIs.* Retrieved May 21, 2020, from Google Identity Platform: https://developers.google.com/identity/protocols/oauth2

HAPI FHIR. (2020). *HAPI FHIR - The Open Source FHIR API for Java*. Retrieved January 10, 2020, from https://hapifhir.io/

Hardt, D. (2012, October). *The OAuth 2.0 Authorization Framework.* Retrieved from Internet Engineering Task Force (IETF): https://tools.ietf.org/html/rfc6749

HealthIT.gov. (2014, May 12). *What is HIE?* Retrieved from HealthIT: http://www.healthit.gov/providers-professionals/health-information-exchange/what-hie

HL7 International. (2013, 08 01). *Value Sets using code system: Confidentiality*. Retrieved June 27, 2020, from HL7 International: http://www.hl7.org/documentcenter/public/standards/vocabulary/vocabulary_tables/infrastructure/vocabulary/vs_Confidentiality.html

HL7 International. (2019, January 4). *Health Level Seven International*. Retrieved from https://www.hl7.org/

HL7 International. (2019, November 1). *Resourcelist - FHIR v4.0.1*. Retrieved January 19, 2020, from https://www.hl7.org/fhir/resourcelist.html

HL7 International. (2019, November 1). *Resourcelist - FHIR v4.0.1*. Retrieved January 19, 2020, from https://www.hl7.org/fhir/resourcelist.html

HL7 International. (2019, 3 27). *Security - FHIR v4.0.0.* Retrieved from FHIR: https://www.hl7.org/fhir/security.html#binding

HL7 International. (2019). *USCore*. Retrieved July 2, 2020, from HL7 International: https://www.hl7.org/fhir/us/core/ValueSet-us-core-careteam-provider-roles.html

HL7 International. (2020, May 9). *Overview*. Retrieved from HL7 FHIR: https://www.hl7.org/fhir/overview.html

Housley, R., Polk, W., Ford, W., & Solo, D. (2002, April). *Internet X.509 Public Key Infrastructure Certificate and CRL Profile*. Retrieved from The Internet Engineering Task Force: http://www.ietf.org/rfc/rfc3280.txt

Hu, V. C., Ferraiolo, D., Kuhn, R., Schnitzer, A., Sandlin, K., Miller, R., & Scarfone, K. (2014). *Guide to Attribute Based Access Control (ABAC) Definition and Considerations.* NIST Special Publication. doi:http://dx.doi.org/10.6028/NIST.SP.800-162

Identity Theft Resource Center. (2020). *2019 End of Year Data Breach Report.* Retrieved May 20, 2020, from https://www.idtheftcenter.org/wp-content/uploads/2020/01/01.28.2020_ITRC_2019-End-of-Year-Data-Breach-Report_FINAL_Highres-Appendix.pdf

Kelly, V. (2013, Dec.). *Global Healthcare Stakeholders Want Standards-Based Interoperability and Communications, According to IEEE*. Retrieved from IEEE Standards Association: https://standards.ieee.org/news/2013/ieeesa_mhealth-summit.html

Legion of the Bouncy Castle Inc. (2013). Retrieved July 5, 2020, from The Legion of the Bouncy Castle: https://www.bouncycastle.org/

Lewis, N. (2011, 10 21). *80% Of Doctors Use Mobile Devices At Work*. Retrieved May 9, 2020, from Information Week: https://www.informationweek.com/mobile/80--of-doctors-use-mobile-devices-at-work/d/d-id/1100880?

Mavridis, I., Georgiadis, C., Pangalos, G., & Khair, M. (2001, Jan-Mar). Access Control based on Attribute Certificates for Medical Intranet Applications. *Journal of Medical Internet Research, 3*(1).

Mettler, T., & Rohner, P. (2009). Increasing the Networkability of Health Service Providers: The Case of Switzerland. *Sprouts: Working Papers on Information Systems, 9*(1).

Montopoli, B. (2013, August 7). *For criminals, smartphones becoming prime targets*. (CBS News) Retrieved 10 30, 2016, from http://www.cbsnews.com/news/for-criminals-smartphones-becoming-prime-targets/

NUCC. (2020). *Provider Taxonomy*. Retrieved July 2, 2020, from National Uniform Claim Committee: http://www.nucc.org/index.php/code-sets-mainmenu-41/provider-taxonomy-mainmenu-40

OpenEMR. (2020, May 21). *OpenEMR*. Retrieved from OpenEMR: https://www.open-emr.org/

OpenSSL Software Foundation. (2018). *index*. Retrieved from OpenSSL Cryptography and SSL/TLS Toolkit: https://www.openssl.org/

Oracle. (2013). *What is a servlet?* Retrieved from https://docs.oracle.com/javaee/6/tutorial/doc/bnafe.html

Rivera Sánchez, Y. (2017). *A Configurable Framework for RBAC, MAC, and DAC for Mobile Applications.* Storrs: Doctoral Dissertations. Retrieved from https://opencommons.uconn.edu/dissertations/1572/

Rivera Sánchez, Y. (2017). *A Configurable Framework for RBAC, MAC, and DAC for Mobile Applications.* Storrs.

Ryutov, T., Zhou, L., Neuman, C., Leithead, T., & Seamons, K. (2005). Adaptive Trust Negotiation and Access Control. *SACMAT '05 Proceedings of the tenth ACM symposium on Access control models and technologies* (pp. 139-146). New York City: ACM New York, NY, USA ©2005.

Sanzi et al. (2017). Integrating Trust Profiles, Trust Negotiation, and Attribute Based Access Control. *2017 5th IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud)* (pp. 177-184). San Francisco: IEEE. doi:10.1109/MobileCloud.2017.30

Sanzi, E., & Demurjian, S. (May 2016). Identification and Adaptive Trust Negotiation in Interconnected Systems. In A. Malik, A. Anjum, & B. Raza (Eds.), *Innovative Solutions for Access Control Management* (pp. 33-65). IGI Global.

Sanzi, E., Demurjian, S., & Billings, J. (2017). Integrating Trust Profiles, Trust Negotiation, and Attribute Based Access Control. *2017 5th IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud)* (pp. 177-184). San Francisco: IEEE. doi:10.1109/MobileCloud.2017.30

Sanzi, E., Demurjian, S., Agresta, T., & Murphy, A. (November 2016). Trust Profiling to Enable Adaptive Trust Negotiation in Mobile Devices. In S. Mukherja (Ed.), *Mobile Application Development, Usability, and Security* (pp. 95-116). IGI Global.

SNOMED International. (2020). *SNOMED*. Retrieved July 2, 2020, from 5-Step Briefing: https://www.snomed.org/snomed-ct/five-step-briefing

Sundelin, T. L. (July 2003). *Surrogate Trust Negotiation: Solving Authentication and Authorization Issues in Dynamic Mobile Networks.* Bringham Young University.

The Apache Software Foundation. (2018). *Welcome to the Apache Struts project*. Retrieved July 5, 2020, from Struts: https://struts.apache.org/index.html

The Apache Software Foundation. (2020). *Welcome!* Retrieved July 5, 2020, from Apache Tomcat: http://tomcat.apache.org/

The Office of the National Coordinator for Health Information Technology. (2018, September 19). *Meaningful Consent Overview | HealthIT.gov*. Retrieved January 24, 2020, from https://www.healthit.gov/topic/meaningful-consent-overview

The Office of the National Coordinator for Health Information Technology. (2019, April 17). *Patient Consent for Electronic Health Information Exchange | HealthIT.gov*. Retrieved January 24, 2020, from https://www.healthit.gov/topic/patient-consent-electronic-health-information-exchange

Twitter, Inc. (2020). *Twitter Developers.* Retrieved May 21, 2020, from https://developer.twitter.com/en/docs/basics/authentication/overview

U.S. Department of Health & Human Services. (2019, 01 04). *Health Information Privacy*. Retrieved 05 9, 2020, from hhs.gov: https://www.hhs.gov/hipaa/index.html

Vawdrey, D. K., Sundelin, T. L., Seamons, K. E., & Knutson, C. D. (2003). Trust Negotiation for Authentication and Authorization in Healthcare Information Systems. *Engineering in Medicine and Biology Society, 2003. Proceedings of the 25th Annual International Conference of the IEEE* (pp. 1406-1409). IEEE.

Ventola, C. L. (2014, May). Mobile Devices and Apps for Health Care Professionals: Uses and Benefits. *Pharmacy and Therapeutics, 39*(5), 356-364. Retrieved from http://www.ncbi.nlm.nih.gov/pmc/articles/PMC4029126

W3C. (2007). *SOAP Specifications*. Retrieved from https://www.w3.org/TR/soap/

Winsborough, W. H., Seamons, K. E., & Jones, V. E. (2000). Automated trust negotiation. *DARPA Information Survivability Conference and Exposition, 2000. DISCEX '00. Proceedings. 1*, pp. 88 - 102. Hilton Head, SC: IEEE. doi:10.1109/DISCEX.2000.824965

WorldVistA. (2020, May 21). Retrieved from WorldVistA: http://worldvista.org/

Yale New Haven Health; Yale Medical Group. (2020, May 21). *FAQ Care Everywhere.* Retrieved from YaleNewHavenHealth: https://projectepic.ynhh.org/Epic%20Newsletters%20and%20Fact%20Sheets/FAQ%20Care%20Everywhere.pdf